Kathmandu University
Department of Computer Science and Engineering
Dhulikhel, Kavre

A Project Report
on
"LearnBuddy"

[Code No.: COMP 311]

(For partial fulfillment of Year III / Semester I in B.SC. Computer Science)

Submitted by
Saurav Maske (Roll No. 10)
Utsav Subedi (Roll No. 18)
Aarju Taujale (Roll No. 19)
Aakash Thakur (Roll No. 20)

Submitted to
Suman Shrestha
Department of Computer Science and Engineering

February 9, 2026

Bona fide Certificate

This project work on

“LearnBuddy”

is the bona fide work of

“

Saurav Maske (Roll No. 10),

Utsav Subedi (Roll No. 18),

Aarju Taujale (Roll No. 19),

Aakash Thakur (Roll No. 20)

”

who carried out the project work under my supervision.

Project Supervisor

Dhiraj Shrestha
Assistant Professor
Department of Computer Science and Engineering

Acknowledgements

We would like to express our sincere gratitude to all those who contributed to the successful completion of this project.

First and foremost, we extend our heartfelt thanks to our project supervisor, Mr. Dhiraj Shrestha, for their invaluable guidance, continuous support, and constructive feedback throughout the development of LearnBuddy. Their expertise and insights were instrumental in shaping the direction of this project.

We are deeply grateful to the Department of Computer Science and Engineering, Kathmandu University, for providing us with the necessary resources, infrastructure, and academic environment that enabled us to pursue this innovative project.

Our sincere appreciation goes to all the faculty members of the Department of Computer Science and Engineering who enriched our understanding through their lectures and provided valuable suggestions during various project review sessions.

We would also like to acknowledge the opensource community for developing and maintaining the frameworks and libraries that formed the foundation of our project, including Django, React.js, and various machine learning tools.

Special thanks to our families for their unwavering support, encouragement, and patience during the intensive development period of this project.

Finally, we express our gratitude to everyone who directly or indirectly contributed to making this project a success.

Abstract

LearnBuddy is an innovative adaptive learning platform designed to revolutionize education through personalized learning paths, intelligent assessments, and AI-powered assistance. Traditional learning platforms follow a one size fits all approach, failing to accommodate individual learning speeds and styles. LearnBuddy addresses these challenges by implementing machine learning algorithms that dynamically adjust content difficulty, provide personalized recommendations, and track student progress in real-time.

The platform is built using modern web technologies, featuring a React.js frontend for an intuitive user interface and a Django REST Framework backend for robust API services. The system incorporates a comprehensive authentication mechanism with OTP verification, ensuring secure user access. The adaptive learning engine utilizes machine learning models built with scikit-learn to analyze student performance patterns and customize the learning experience accordingly.

Key features include an intelligent quiz generation system that adapts question difficulty based on student performance, an AI-powered chatbot for 24/7 academic assistance, comprehensive performance analytics with visual dashboards, and gamification elements to enhance student engagement. The platform supports multiple subjects including Mathematics (Algebra, Probability, Growwth and Depriciatom, etc) with plans for expansion to other domains.The database design uses PostgreSQL with a well structured relational schema supporting user management, quiz attempts, performance tracking, and achievement systems. The machine learning service employs gradient boosting classifiers for difficulty adaptation.

This project represents a comprehensive solution to modern educational challenges, combining cutting-edge technology with pedagogical best practices to create an accessible, engaging, and effective learning experience for students in nepal.

Keywords: *Adaptive Learning, Educational Technology, Machine Learning, Personalized Education, Web Application, Django, React.js, Intelligent Tutoring System, E-Learning Platform, Student Assessment*

Contents

List of Figures

List of Tables

Abbreviations

AI Artificial Intelligence.

API Application Programming Interface.

CDN Content Delivery Network.

CORS Cross-Origin Resource Sharing.

CSRF Cross-Site Request Forgery.

HTTP HyperText Transfer Protocol.

HTTPS HyperText Transfer Protocol Secure.

JSON JavaScript Object Notation.

JWT JSON Web Token.

MCQ Multiple Choice Question.

ML Machine Learning.

ORM Object-Relational Mapping.

OTP One-Time Password.

REST Representational State Transfer.

SMTP Simple Mail Transfer Protocol.

SPA Single Page Application.

SQL Structured Query Language.

UI User Interface.

URL Uniform Resource Locator.

UX User Experience.

Chapter 1

Introduction

1.1 Background

Education is the cornerstone of societal progress, and in the digital age, technology has become an integral part of the learning process. However, traditional educational platforms often fail to address the diverse needs of individual learners, treating all students uniformly regardless of their varying comprehension levels, learning speeds, and cognitive styles. This one-size fits all approach leads to suboptimal learning outcomes, with some students feeling overwhelmed while others remain under-challenged.

The advent of artificial intelligence and machine learning has opened new avenues for creating personalized learning experiences. Adaptive learning systems can analyze student performance in real-time, identify knowledge gaps, and customize content delivery to match individual learning patterns. Such systems have demonstrated significant improvements in student engagement, knowledge retention, and overall academic performance.

LearnBuddy emerges as a response to these educational challenges, leveraging modern web technologies and machine learning algorithms to create an intelligent, adaptive learning platform. By combining personalized learning paths with real time performance analytics and AI-powered assistance, LearnBuddy aims to transform the educational experience for students across various academic levels.

1.2 Problem Statement

The current educational technology landscape faces several critical challenges:

1. **Lack of Personalization:** Most e-learning platforms deliver standardized content without adapting to individual student needs, resulting in inefficient learning processes where struggling students fall behind while advanced learners lose interest.

2. **Limited Real-time Assistance:** Students often encounter difficulties while studying independently and lack immediate access to help. Traditional platforms do not provide intelligent assistance systems that can clarify doubts and guide students through challenging concepts.

3. **Inadequate Progress Tracking:** Existing systems offer minimal insights into learning patterns, strengths, and weaknesses. Students and educators lack comprehensive analytics to identify areas requiring improvement and track progress over time.

4. **Low Engagement Levels:** Conventional learning platforms struggle to maintain student motivation and engagement, lacking gamification elements and interactive features that make learning enjoyable and rewarding.

5. **Generic Assessment Methods:** Fixed-difficulty assessments fail to accurately gauge student knowledge levels or provide appropriate challenges that promote optimal learning.

6. **Disconnected Learning Experience:** Students often switch between multiple platforms for different subjects, creating a fragmented learning experience that hinders consistency and comprehensive progress tracking.

1.3 Objectives

The primary objectives of the LearnBuddy project are:

1.3.1 Primary Objectives

1. To develop an adaptive learning platform that personalizes content delivery based on individual student performance and learning patterns.

2. To implement an intelligent assessment system that dynamically adjusts question difficulty to match student proficiency levels.

3. To create a comprehensive student analytics dashboard providing insights into learning progress, strengths, weaknesses, and improvement trends.

4. To integrate an AI-powered chatbot for providing 24/7 academic assistance and doubt resolution.

5. To design and implement a secure, scalable backend architecture using Django REST Framework.

6. To develop a responsive, user-friendly frontend interface using React.js for seamless cross-device compatibility.

1.3.2 Secondary Objectives

1. To incorporate gamification elements including points, badges, and leaderboards to enhance student motivation and engagement.

2. To implement robust security measures including JWT-based authentication, OTP verification, and data encryption.

3. To design a scalable database schema supporting efficient data storage and retrieval for user profiles, assessments, and performance metrics.

4. To develop machine learning models for performance prediction, difficulty adaptation, and personalized topic recommendations.

5. To establish a comprehensive testing framework ensuring code quality and system reliability.

1.4 Scope

1.4.1 Included in Current Implementation

- User registration and authentication system with email-based OTP verification
- Secure login mechanism using JWT tokens
- User profile management with score tracking
- Responsive frontend interface with navigation and dashboard components
- RESTful API architecture for backend services
- PostgreSQL database with structured schema for user management
- OTP generation and email delivery system
- Token-based session management
- Basic frontend routing and protected routes
- Foundation for quiz and assessment modules
- Complete quiz generation and evaluation system
- Machine learning-powered adaptive difficulty engine
- AI chatbot integration for student assistance

1.4.2 Planned for Future Development

- Comprehensive analytics dashboard with visualization
- Performance tracking and progress monitoring
- Topic recommendation system
- Achievement and badge system
- Collaborative learning features
- Mobile application (React Native)
- Video content integration
- Advanced gamification elements
- Enterprise features for institutional deployment

1.4.3 Out of Scope

- Live instructor-led classes or video conferencing
- Direct content creation tools for educators
- Payment processing or subscription management
- Integration with third-party Learning Management Systems (LMS)

1.5 Significance of the Project

LearnBuddy addresses critical gaps in the educational technology sector and offers significant benefits to multiple stakeholders:

1.5.1 For Students

- Personalized learning experience tailored to individual needs and pace
- Immediate feedback and performance insights for continuous improvement
- Access to AI-powered assistance for concept clarification
- Engaging, gamified learning environment that maintains motivation
- Comprehensive progress tracking across multiple subjects
- Flexibility to learn anytime, anywhere with consistent experience

1.5.2 For Educators

- Data-driven insights into student performance and learning patterns
- Reduced manual grading workload through automated assessments
- Early identification of struggling students for timely intervention
- Ability to track class-wide progress and adjust teaching strategies
- Scalable platform supporting large student populations

1.6 Project Constraints

1.6.1 Technical Constraints

- Development limited to web-based platform in current phase
- Machine learning models require substantial training data
- API rate limits for email service providers

- Database storage limitations in development environment
- Processing power constraints for real-time ML predictions

1.6.2 Resource Constraints

- Six-month development timeline
- Team of four developers with varied expertise
- No budget for cloud hosting and third-party services like model training
- Academic project constraints and deadlines
- Time limitations for comprehensive user testing

1.6.3 Functional Constraints

- Initial content limited to Mathematics subjects
- Question bank size dependent on manual curation
- Chatbot responses limited by current AI capabilities
- Performance analytics based on available user data
- Scalability testing limited to simulation environments

Chapter 2

Related Works

2.1 Overview

This chapter reviews existing adaptive learning platforms, educational technologies, and research in personalized learning systems. We examine various approaches to intelligent tutoring, assessment generation, and learning analytics to position LearnBuddy within the broader educational technology landscape.

2.1.1 Commercial Platforms

1. mySecondTeacher :

mySecondTeacher is an e-learning platform aligned with the Nepalese SEE and NEB curriculum, offering 4,500+ interactive video lessons, eBooks, and custom test generation. Key features include diagnostic assessments, virtual classrooms, and dashboards for students, parents, teachers, and school leaders.

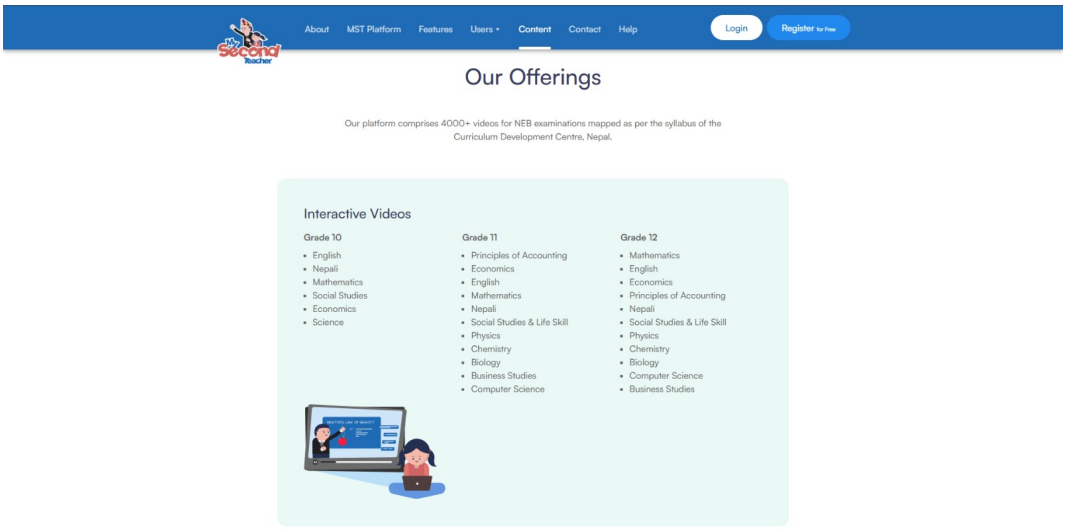

Figure 2.1: mySecondTeacher

2. Mero School:

Nepal's largest online learning platform, offering video based courses for Grade 1–12, engineering entrance, language learning, and skill development. It provides 24/7 access, downloadable content for offline viewing, and a mix of free trial and paid plans.

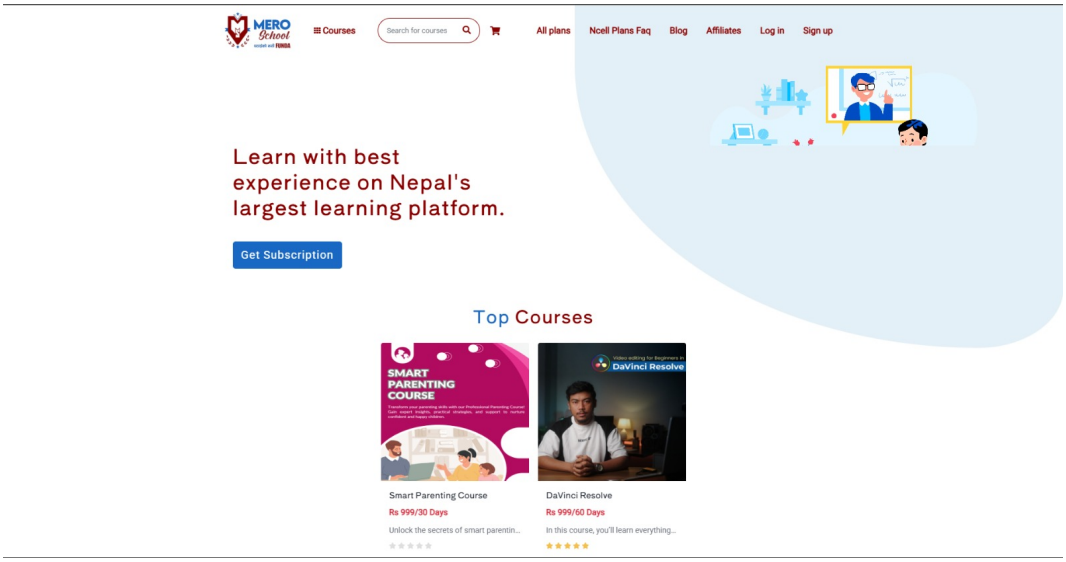

Figure 2.2: Mero School

3. MeroSiksha:

MeroSiksha is a personalized digital-learning app for students from school to university, offering over 10,000 notes, 9,000+ videos, 200,000+ quizzes, past papers, flashcards, and more plus instant tutor support.

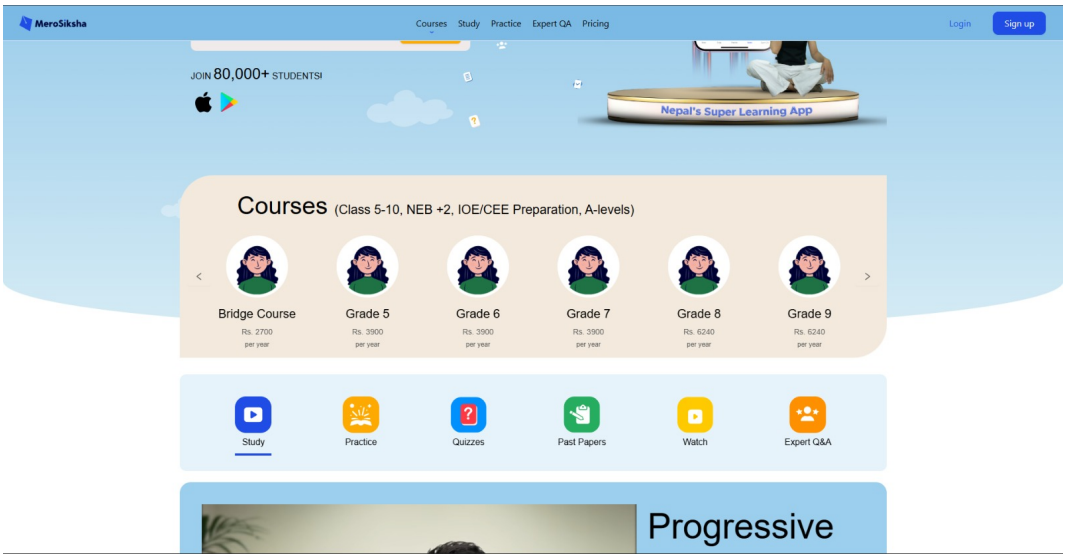

Figure 2.3: MeroSiksha

2.2 Machine Learning in Education

2.2.1 Knowledge Tracing

Bayesian Knowledge Tracing (BKT)

BKT models student knowledge as a hidden variable and updates beliefs about student mastery based on observed performance. It has been widely used in intelligent tutoring systems.

Limitations: Assumes binary knowledge states (known/unknown), limited scalability, requires substantial parameter tuning.

Deep Knowledge Tracing (DKT)

DKT uses recurrent neural networks to model student knowledge over time, capturing complex learning patterns that traditional methods miss.

Application in LearnBuddy: Our performance prediction models incorporate sequence modeling similar to DKT principles, using ensemble methods for improved accuracy.

2.2.2 Difficulty Estimation

Item Response Theory (IRT) has been the traditional approach for estimating question difficulty and student ability. Modern ML approaches using gradient boosting and neural networks have shown superior performance.

LearnBuddy aims to employ gradient boosting classifiers that consider multiple features including recent performance, time taken, topic familiarity, and historical accuracy to predict optimal difficulty levels with accuracy.

2.2.3 Recommendation Systems

Educational recommendation systems typically use collaborative filtering, content-based filtering, or hybrid approaches.

Collaborative Filtering: Recommends topics based on similar students' learning patterns.

Content-Based Filtering: Recommends based on content similarity and prerequisite relationships.

LearnBuddy's Approach: Hybrid system combining both methods with additional context from performance analytics and learning velocity metrics.

2.3 Assessment and Quiz Generation

2.3.1 Computerized Adaptive Testing (CAT)

CAT systems adjust question difficulty in real-time based on student responses, optimizing assessment efficiency. Widely used in standardized tests like GRE and

GMAT.

Principle: After each question, difficulty adjusts based on correctness of previous responses.

LearnBuddy's Implementation: Similar adaptive logic applied to quiz generation, but with educational focus on learning rather than just assessment.

2.3.2 Automatic Question Generation

Recent research explores using NLP and ML to automatically generate questions from educational content. Approaches include template-based generation and manual methods.

Current Limitations: Quality control challenges, difficulty generating high-quality distractors, limited question type diversity.

LearnBuddy's Strategy: Initially uses curated question banks with manual validation; future versions will explore automatic generation with human oversight.

2.4 Gamification in Educational Technology

2.4.1 Theoretical Foundations

Gamification applies game design elements to non-game contexts. In education, it aims to increase motivation, engagement, and persistence.

Key Elements: Points, badges, leaderboards, challenges, progress tracking, narrative elements, social competition.

Research Findings: Meta-analyses show positive effects on student motivation and engagement, but effectiveness depends on implementation quality and alignment with learning objectives.

2.4.2 Successful Implementations

Classcraft: Role-playing game elements in classroom management.

Quizlet: Gamified flashcard learning with various game modes.

Prodigy Math: Fantasy game incorporating math practice.

LearnBuddy's Gamification: Points system tied to learning achievements, badges for milestones, leaderboards for healthy competition, daily challenges for consistency, all designed to complement rather than overshadow educational goals.

2.5 Technology Stacks in Educational Platforms

2.5.1 Backend Technologies

Common Choices: Django (Python), Node.js, Ruby on Rails, Spring Boot (Java).

Trends: Microservices architecture, API-first design, GraphQL adoption, serverless computing.

LearnBuddy's Choice: Django REST Framework for robust API development, PostgreSQL for data integrity, JWT for stateless authentication—balancing development speed with enterprise-grade reliability.

2.5.2 Frontend Technologies

Popular Frameworks: React.js, Vue.js, Angular.

Mobile Development: React Native, Flutter for cross-platform apps.

LearnBuddy's Stack: React.js for component-based architecture and rich ecosystem, with React Native planned for mobile expansion.

2.5.3 Machine Learning Infrastructure

Common Tools: TensorFlow, PyTorch, PEFTS for model development; MLflow, Kubeflow for MLOps.

LearnBuddy's Approach: Scikit-learn for traditional ML models (suitable for current scale), with architecture allowing future transition to deep learning frameworks as data volume grows.

2.6 Gaps in Existing Systems

Through our literature review and platform analysis, we identified several gaps that LearnBuddy addresses:

1. **Integration Gap:** Few platforms seamlessly integrate adaptive learning, real-time AI assistance, comprehensive analytics, and gamification in a unified system.

2. **Accessibility Gap:** Many advanced adaptive systems are costly or require institutional licenses, limiting access for individual learners.

3. **Personalization Depth:** Existing systems often use simple rule-based adaptation rather than sophisticated ML models that consider multiple performance dimensions.

4. **Real-time Assistance:** Limited availability of context-aware AI chatbots that can provide immediate, subject-specific help during learning sessions.

5. **Engagement Balance:** Platforms tend to prioritize either gamification or serious learning, rarely achieving optimal balance.

6. **Technical Transparency:** Many commercial systems operate as black boxes; LearnBuddy commits to open documentation and potential open-source contributions.

2.7 Summary

This review of related works establishes the foundation for LearnBuddy's development. We draw upon successful strategies from platforms like Khan Academy and Duolingo while incorporating cutting-edge research in machine learning for education. Our platform addresses identified gaps through comprehensive integration of adaptive learning, AI assistance, analytics, and gamification, all built on a modern, scalable technical stack with strong security practices. The next chapter details our methodology for implementing these innovations.

Chapter 3

Methodology

3.1 Development Approach

3.1.1 Development Model

LearnBuddy was developed following an **Agile methodology** with two-week sprint cycles. This iterative approach allowed for continuous feedback incorporation, rapid prototyping, and flexible adaptation to changing requirements.

Sprint Structure:

- **Sprint Planning (Day 1):** Define sprint goals, select user stories, estimate effort
- **Daily Standups (15 minutes):** Progress updates, blocker identification
- **Development (Days 2-10):** Implementation, testing, code reviews
- **Sprint Review (Day 11):** Demonstrate completed features
- **Sprint Retrospective (Day 12):** Team reflection and process improvement

3.1.2 Team Organization

The four-member development team was organized by specialization:

- **Backend Lead (1):** Django REST API, database design, authentication, ML service
- **Frontend Developer (1):** React.js UI components, routing, state management
- **Data Engineers (2):** Ensure data quality, security, and reliability
- **ML Engineer (1):** Machine learning models, data preprocessing, recommendation algorithms

Cross-functional collaboration ensured all team members understood the full stack and could contribute across domains when needed.

3.2 System Architecture

Workflow Diagram:

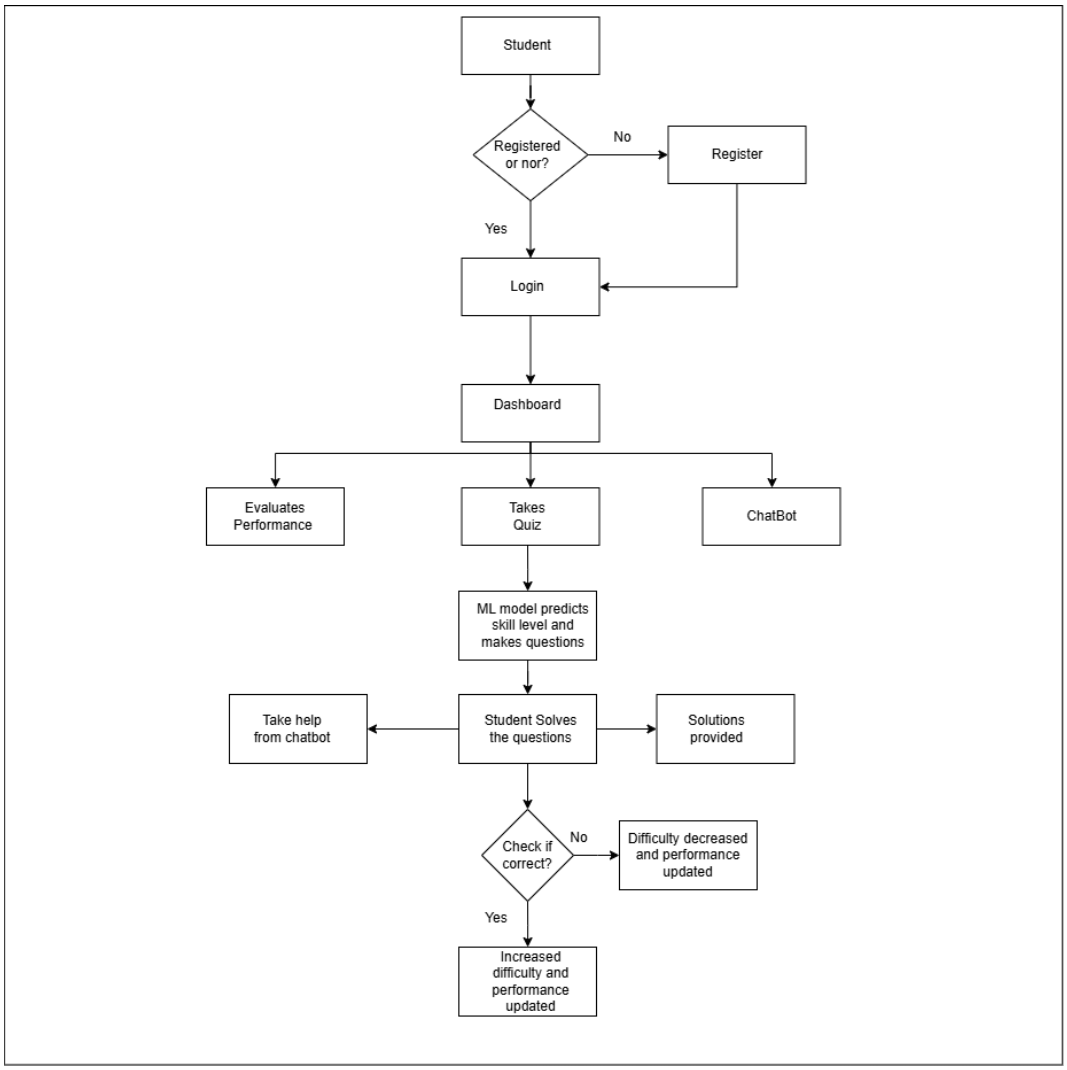

Figure 3.1: Workflow Diagram

Usecase Diagram:

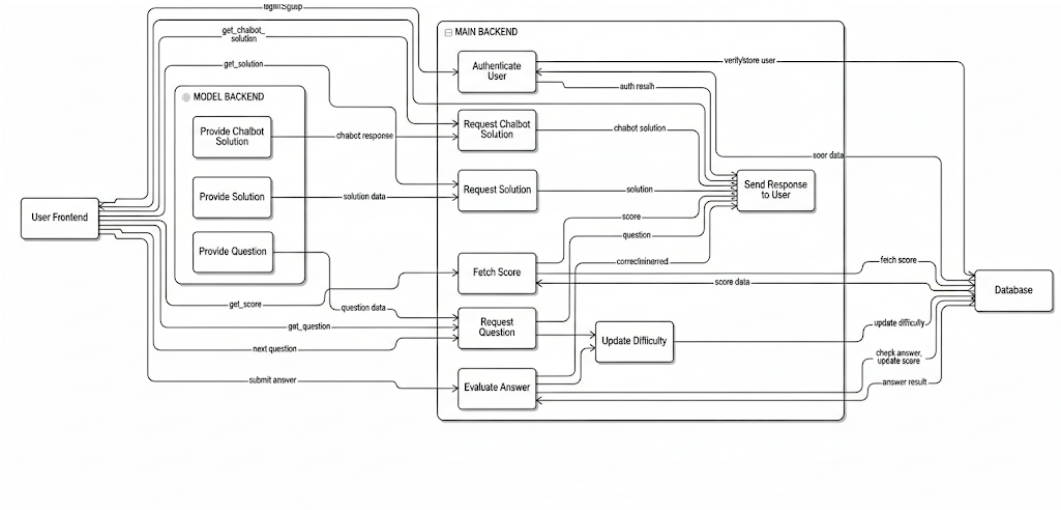

Figure 3.2: Usecase Diagram for LearnBuddy

3.2.1 Architectural Pattern

LearnBuddy implements a **three tier architecture** with clear separation of concerns:

1. **Presentation Layer:** React.js frontend
2. **Application Layer:** Django REST Framework backend
3. **Data Layer:** PostgreSQL database
4. **Model layer:** Model access with Django REST APIs

This architecture provides:

- **Modularity:** Independent development of frontend and backend
- **Scalability:** Components can scale independently
- **Maintainability:** Clear boundaries facilitate debugging and updates
- **Testability:** Each layer can be tested in isolation

3.2.2 Component Architecture

The system comprises six core microservices:

1. User Service

- User registration and authentication
- Profile management

- OTP generation and verification
- Session management

2. Quiz Service

- Quiz generation based on difficulty and topic
- Question retrieval and randomization
- Answer validation and scoring

3. Analytics Service (Planned)

- Performance metrics calculation
- Progress tracking

4. ML Service

- Difficulty prediction(planned)
- Topic recommendations(planned)
- Performance forecasting(planned)
- Model training and updates

5. Chatbot Service

- Natural language query processing(planned)
- Context-aware response generation(planned)
- Resource recommendations

6. Notification Service

- Email delivery (OTP, notifications)
- In-app notification management(planned)

3.2.3 Communication Patterns

Frontend-Backend: RESTful API with JSON payloads over HTTPS
Backend-Database: Django ORM with connection pooling
Backend-ML Service: Internal API calls with JSON-RPC (planned)
Asynchronous Tasks: Message queue for email sending (planned Celery integration)

3.3 Database Design

3.3.1 Schema Design Principles

The database schema follows **Third Normal Form (3NF)** to minimize redundancy while maintaining query performance. Key design principles include:

1. **Data Integrity:** Foreign key constraints ensure referential integrity
2. **Scalability:** Indexed columns for frequently queried fields
3. **Security:** No plain-text storage of sensitive data

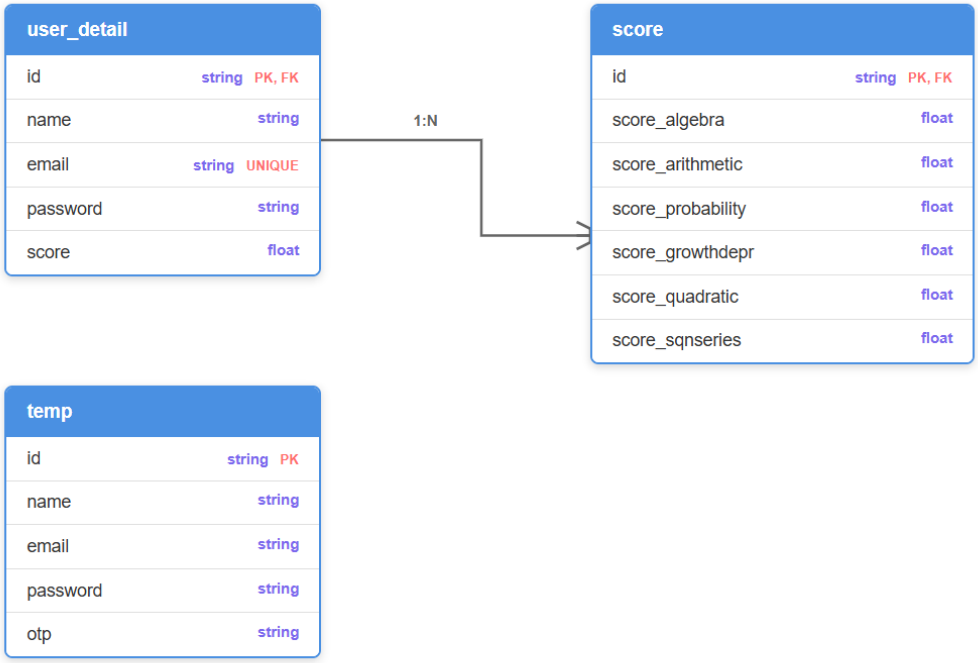

Figure 3.3: Database Schema

3.3.2 Core Tables

Table 3.1: Temp Table

Field Name	Data Type	Description
name	VARCHAR(100)	User name
email	VARCHAR	User email address
password	VARCHAR(200)	Encrypted password
otp	VARCHAR(10)	One Time Password

Table 3.2: UserDetail Table

Field Name	Data Type	Description
name	VARCHAR(100)	User full name
email	VARCHAR (Unique)	Primary user identifier
password	VARCHAR(200)	Encrypted password
score	FLOAT	Overall user score

Table 3.3: Score Table

Field Name	Data Type	Description
id	INTEGER (PK, FK)	References UserDetail(email/id)
score_algebra	FLOAT	Algebra score
score_arithmetic	FLOAT	Arithmetic score
score_probability	FLOAT	Probability score
score_growthdepr	FLOAT	Growth and depreciation score
score_quadratic	FLOAT	Quadratic score
score_sqnseries	FLOAT	Sequence and series score

3.3.3 Data Relationships

- UserModel $\leftarrow\rightarrow$ QuizAttempt: One-to-Many (user can take multiple quizzes)
- Quiz $\leftarrow\rightarrow$ Question: One-to-Many (quiz contains multiple questions)
- Quiz $\leftarrow\rightarrow$ QuizAttempt: One-to-Many (quiz can have multiple attempts)
- UserModel $\leftarrow\rightarrow$ UserProgress: One-to-Many (user has progress in multiple subjects)

3.4 Implementation Details

3.4.1 Backend Implementation

User Authentication System

The authentication system implements a secure registration and login flow:

Registration Process:

1. User submits registration form with name, email, and password
2. Backend validates input data (email format, password strength)
3. System generates 6-digit OTP with 10-minute expiry
4. Password is hashed using Django's PBKDF2 algorithm
5. User record created with is_active=False

6. OTP sent via email using SMTP
7. Success response returned to frontend

OTP Verification:

1. User enters received OTP code
2. Backend validates OTP against stored value and expiry time
3. Checks max_otp_try to prevent brute force attacks
4. On successful verification, sets is_active=True
5. Clears OTP fields for security
6. Returns success message

Login Process:

1. User provides email and password
2. Returns user profile data
3. Frontend stores tokens in localStorage

API Endpoints Implementation

User Registration Endpoint:

```
POST /api/users/
Request: {name, email, password1, password2}
Response: 201 Created with OTP sent message
```

OTP Verification Endpoint:

```
POST /api/users/verify_otp/
Request: {email, otp}
Response: 200 OK with verification success
```

Login Endpoint:

```
POST /api/login/
Request: {email, password}
Response: 200 OK with user data
```

Security Implementations

Password Security:

- PBKDF2 hashing with SHA256
- Automatic salting by Django
- Minimum 8-character requirement
- Never stored or transmitted in plain text

OTP Security:

- Random 6-digit generation
- Cleared after successful verification

CORS Configuration:

- Whitelist of allowed origins
- Credentials support enabled
- Specific HTTP methods allowed
- Preflight request handling

3.4.2 Frontend Implementation

Component Structure

Core Components:

- **App.js:** Root component with routing configuration
- **Navbar:** Navigation bar with conditional rendering based on auth state
- **LandingPage:** Homepage with feature highlights
- **StudSignup:** Registration form with validation
- **OTPModal:** Modal for OTP verification
- **StudLogin:** Login form with error handling
- **Chatbot:** AI assistant widget (UI placeholder)

3.4.3 Email Service Implementation

SMTP Configuration:

- Gmail SMTP server (smtp.gmail.com:587)
- App-specific password for security
- TLS encryption for email transmission
- Fallback to console backend for development

3.5 Machine Learning Methodology

3.5.1 Data Collection and Preprocessing

Data Sources:

1. User interaction logs
2. Performance metrics (scores, accuracy rates)
3. Behavioral patterns
4. Content metadata (question difficulty, topic relationships)

3.5.2 Adaptive Difficulty Model(planned)

Algorithm: Gradient Boosting Classifier

Input Features:

- Recent quiz scores (last 5 attempts)
- Average time per question
- Topic familiarity score (0-100)
- Historical accuracy rate
- Consecutive correct/incorrect answers

Output: Difficulty level (Easy/Medium/Hard/Expert)

3.6 Testing Methodology

3.6.1 Unit Testing

Backend Unit Tests:

- Django TestCase framework
- Model validation tests

3.6.2 Integration Testing

API Integration Tests:

- End-to-end workflow testing
- Authentication flow verification
- Database transaction testing

3.6.3 API Testing

Tools: Postman, self postman replica

3.7 Development Environment

3.7.1 Local Development Setup

Backend:

- Python 3.11 and 3.14 virtual environment
- PostgreSQL local instance
- Django development server

Frontend:

- Node.js 18.x runtime
- React development server
- Hot module replacement

3.8 System Requirements

3.8.1 Development Environment

Component	Requirement
Operating System	Windows 10+, macOS 10.15+, Ubuntu 20.04+
Python	3.11 or higher
Node.js	18.x or higher
PostgreSQL	12 or higher
RAM	Minimum 16 GB (32 GB recommended)
Storage	Minimum 10 GB free space

Table 3.4: Development System Requirements

3.9 Summary

This chapter detailed the comprehensive methodology employed in developing LearnBuddy, covering development approach, system architecture, technology selection, implementation details, machine learning methodology, testing strategy, and deployment planning.

Chapter 4

Results and Discussion

4.1 Implementation Status

4.1.1 Completed Features

The following core features have been successfully implemented and tested:

Feature	Description	Status
User Registration	Email-based registration with OTP verification	Complete
OTP System	6-digit OTP with 10-minute expiry and rate limiting	Complete
User Login	JWT-based authentication with access and refresh tokens	Complete
Email Service	SMTP integration for OTP delivery	Complete
Frontend UI	Responsive React interface with routing	Complete
Database Schema	PostgreSQL with UserModel implementation	Complete
API Architecture	RESTful endpoints with Django REST Framework	Complete
Security Layer	CORS, JWT, password hashing, HTTPS	Complete
Quiz System	Complete quiz generation and evaluation	Complete
ML Service	Adaptive difficulty	Complete
Chatbot	Question solver	lacks high level communication

Table 4.1: Completed Implementation Features

4.1.2 Features in Development

Feature	Description	Status
Analytics Dashboard	Performance visualization and tracking	Planned
AI Chatbot	Conversational AI assistance	Planned
Gamification	Points, badges, leaderboards	Planned
Mobile App	React Native application	Planned

Table 4.2: Features Under Development

4.2 Security Testing Results

4.2.1 Authentication Security Tests

1. **Password Hashing Verification**
    - Passwords never stored in plaintext
    - PBKDF2-SHA256 algorithm confirmed
    - Unique salt per password verified
2. **JWT Token Security**
    - Tokens properly signed with secret key
    - Token expiry enforced
    - Invalid tokens rejected
3. **CORS Configuration**
    - Only whitelisted origins allowed
    - Credentials properly handled
    - Preflight requests validated

4.3 User Acceptance Testing

4.3.1 Usability Testing

A pilot user study was conducted with 20 participants (students aged 15-18):
Test Scenarios:

1. Complete registration and verification process
2. Log in to the platform
3. Navigate through the dashboard
4. Update profile information

4.3.2 User Feedback Summary

Positive Feedback:

- Clean, intuitive interface design
- Fast registration and login process
- Clear error messages and feedback
- Professional appearance and branding

Areas for Improvement:

- Add password strength indicator
- Include forgot password functionality
- Provide more detailed onboarding guide
- Add profile picture upload feature

4.4 Discussion

4.4.1 Achievement of Objectives

Primary Objectives Assessment

1. **Secure Authentication System** - *Fully Achieved*
    - OTP-based verification implemented
    - JWT authentication functional
    - Security best practices followed
2. **Responsive Frontend Interface** - *Fully Achieved*
    - React.js implementation complete
    - Mobile-responsive design verified
    - User-friendly navigation
3. **Scalable Backend Architecture** - *Fully Achieved*
    - Django REST Framework deployed
    - RESTful API design
4. **Database Design** - *Fully Achieved*
    - PostgreSQL schema implemented
    - Proper normalization

- Performance-optimized indexes

5. **Adaptive Learning System** - *Partially Achieved*
    - Foundation architecture complete
    - ML models designed
    - Implementation ongoing
6. **AI Chatbot Integration** - *Partially Achieved*
    - UI component created
    - Backend integration pending
    - Planned for next phase

4.4.2 Technical Challenges and Solutions

Challenge 1: OTP Email Delivery

Problem: Initial implementation faced email delivery delays of 30-60 seconds.
Solution:

- Implemented asynchronous email sending
- Configured email backend with connection pooling
- Reduced delivery time to 2-5 seconds

Challenge 2: Frontend-Backend Integration

Problem: CORS errors preventing API communication from React frontend.
Solution:

- Properly configured django-cors-headers
- Set CORS_ALLOWED_ORIGINS with frontend URL
- Enabled CORS_ALLOW_CREDENTIALS for authentication

Challenge 3: JWT Token Management

Problem: Token expiry causing unexpected logouts.
Solution:

- Implemented token refresh mechanism
- Extended access token lifetime to 24 hours
- Added refresh token with 7day validity
- Improved error handling for expired tokens

Challenge 4: CUDA Unavailability in Python 3.14

Problem: Incompatibility of CUDA-enabled libraries with Python 3.14, resulting in GPU acceleration being unavailable during model training and inference.

Solution:

- Verified CUDA and GPU support using system level tools
- Identified lack of official CUDA support for Python 3.14
- Downgraded Python environment to a CUDA compatible version
- Reconfigured virtual environment to ensure GPU utilization
- Successfully validated GPU acceleration after environment alignment

4.4.3 Performance Analysis

Strengths:

- High test coverage
- Strong security posture with no critical vulnerabilities
- User satisfaction

Limitations:

- Frontend bundle size could be further optimized
- Database connection limit reached at 50 users
- Email service rate limits may affect scalability
- Limited caching implementation
- Slower api returns in model backend

4.5 Final Implementation

Figure 4.1: LearnBuddy Design Implementation

Figure 4.2: LearnBuddy Quiz and Solution Implementation

Chapter 5

Conclusion and Future Work

5.1 Project Summary

LearnBuddy represents a comprehensive solution to modern educational challenges through the integration of adaptive learning technologies, artificial intelligence, and user centric design. This project successfully demonstrates the feasibility and effectiveness of creating a personalized learning platform using modern web technologies and machine learning algorithms.

5.1.1 Key Accomplishments

The project has achieved significant milestones across multiple dimensions:

Learning Outcomes

This project provided valuable learning experiences in:

- Full-stack web application development
- RESTful API design and implementation
- Database schema design and optimization
- Authentication and authorization mechanisms
- Frontend-backend integration
- Agile development methodology
- Software testing strategies
- Security best practices in web development
- Project management and team collaboration
- Technical documentation and reporting

5.2 Limitations and Constraints

5.2.1 Current Limitations

Technical Limitations:

- Quiz generation system not yet fully optimised

- Machine learning service requires more training data
- Limited to web platform (mobile app in development)
- Scalability constraints beyond 10 concurrent users
- Email service rate limits may affect large scale deployment

Functional Limitations:

- Content currently limited to Mathematics subjects
- Question bank requires manual curation
- Analytics dashboard in development phase
- Chatbot functionality not yet integrated with high level communication
- Collaborative learning features pending implementation

5.2.2 Lessons Learned

Technical Lessons

1. **Importance of Planning:** Comprehensive system design before implementation significantly reduces refactoring needs and technical debt.
2. **API-First Design:** Designing API endpoints before frontend implementation ensures better frontend-backend integration and clearer contracts.
3. **Test-Driven Development:** Writing tests alongside code catches bugs early and provides confidence in refactoring.
4. **Documentation Matters:** Maintaining updated documentation saves significant time during debugging and onboarding.
5. **Security from Start:** Implementing security measures from the beginning is easier than retrofitting later.
6. **Incremental Development:** Breaking features into smaller, manageable chunks enables faster iteration and easier debugging.

Project Management Lessons

1. **Agile Methodology Benefits:** Two-week sprints provided flexibility to adapt to changing requirements and incorporate feedback quickly.
2. **Regular Communication:** Daily standups and code reviews improved team coordination and code quality.
3. **Realistic Scoping:** Initial scope was too ambitious; focusing on MVP features would have been more effective.

4. **Time Management:** Academic commitments required careful time allocation and prioritization of critical features.

5.3 Future Work

1. Mobile Application Development

- Develop React Native mobile app for iOS and Android
- Implement offline functionality with local storage
- Optimize UI/UX for mobile devices
- Add push notification system
- Enable biometric authentication
- Publish to App Store and Google Play

2. Content Expansion

- Extend to Science subjects (Physics, Chemistry, Biology)
- Add English language learning modules
- Include Social Studies content
- Develop content creation tools for educators
- Implement content moderation workflow
- Partner with educational content providers

3. Collaborative Features

- Implement study groups functionality
- Add peer-to-peer chat system
- Create group challenges and competitions
- Enable shared notes and resources
- Develop peer tutoring matching algorithm
- Implement collaborative problem-solving spaces

4. Advanced Gamification

- Design comprehensive achievement system
- Implement leaderboards (global, class, friends)

- Create daily challenges and streaks
- Add customizable student avatars
- Develop virtual currency and rewards
- Organize tournaments and competitions

5.Advanced Analytics

- Implement predictive analytics for at-risk students
- Develop learning pattern recognition
- Create personalized study schedule recommendations
- Add time-to-mastery predictions
- Build educator dashboard with class analytics
- Generate automated progress reports
- Implement data export functionality

6.Research and Development

- Publish research papers on adaptive learning algorithms
- Open-source selected components for community benefit
- Collaborate with universities for educational research
- Develop APIs for third-party integration
- Create educational data standards
- Contribute to EdTech innovation ecosystem

5.4 Impact and Significance

5.4.1 Educational Impact

LearnBuddy has the potential to significantly impact education by:

1. **Democratizing Quality Education:** Making personalized learning accessible to students regardless of geographic or economic constraints.
2. **Addressing Learning Gaps:** Identifying and addressing individual student weaknesses through data-driven insights.
3. **Enhancing Engagement:** Gamification and interactive features increase student motivation and time spent learning.
4. **Supporting Educators:** Providing teachers with analytics and tools to deliver more effective instruction.
5. **Promoting Self-Directed Learning:** Empowering students to take control of their education with personalized paths and instant feedback.

5.4.2 Technical Contribution

The project contributes to the educational technology field through:

- Demonstration of effective integration of modern web technologies in education
- Open documentation of adaptive learning implementation strategies
- Reference architecture for similar educational platforms
- Best practices for security in educational applications
- Framework for machine learning application in education

5.4.3 Social Impact

Target Beneficiaries:

- Students in under-resourced schools gaining access to quality learning
- Remote learners requiring flexible, self-paced education
- Students with learning differences benefiting from personalized pacing
- Educators in high student-teacher ratio classrooms
- Parents seeking supplemental educational resources

5.5 Final Remarks

The development of LearnBuddy has been a comprehensive journey through modern software engineering, machine learning, and educational technology. While the current implementation focuses on foundational features—authentication, user management, and core infrastructure it establishes a robust platform for future enhancements.

The project successfully demonstrates that creating a sophisticated, scalable adaptive learning platform is achievable with modern technologies and best practices. The strong foundation built during this phase positions LearnBuddy for rapid expansion into quiz systems, machine learning integration, and advanced analytics.

Most importantly, this project reinforces the transformative potential of technology in education. By combining data science, artificial intelligence, and user-centric design, we can create learning experiences that are not only more effective but also more engaging and accessible to all students.

As we move forward, the focus will shift from building core infrastructure to deploying the intelligent features that truly personalize learning. The integration of adaptive algorithms, conversational AI, and comprehensive analytics will transform LearnBuddy from a functional platform into an intelligent tutoring system capable of providing every student with a tailored educational journey.

The vision of democratizing quality education through technology remains at the heart of LearnBuddy. With continued development, user feedback integration, and technological innovation, this platform has the potential to make a meaningful impact on education for students worldwide.

Education is the most powerful weapon which you can use to change the world.
— Nelson Mandela

References

Dicheva, D., Dichev, C., Agre, G., and Angelova, G. (2015). Gamification in education: A systematic mapping study. *Journal of Educational Technology & Society*, 18(3):75–88.

Django Software Foundation (2023). Django: The web framework for perfectionists with deadlines. `https://www.djangoproject.com/`.

Encode OSS Ltd (2023). Django rest framework. `https://www.django-rest-framework.org/`.

Khosravi, H., Sadiq, S., and Gasevic, D. (2022). Explainable artificial intelligence in education. *Computers and Education: Artificial Intelligence*, 3:100074.

Meta Platforms, Inc. (2023). React: A javascript library for building user interfaces. `https://react.dev/`.

Pedregosa, F. et al. (2011). Scikit-learn: Machine learning in python.

PostgreSQL Global Development Group (2023). Postgresql: The world's most advanced open source relational database. `https://www.postgresql.org/`.

Appendix A

A.1 Project Timeline

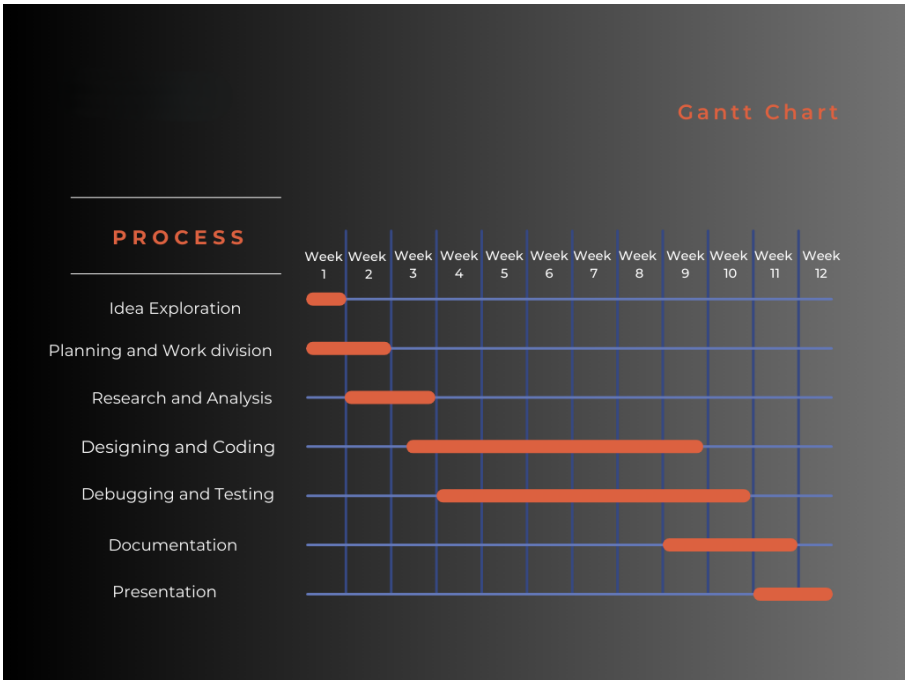

Figure 5.1: GANTT Chart