This project aims to develop a voice-enabled AI chatbot that interacts with students while maintaining full context awareness of the Learning Management System (LMS). The chatbot will provide real-time responses to academic queries, assist with coursework, and offer personalized recommendations based on user activity and progress. By integrating natural language processing (NLP) and speech recognition, the chatbot will ensure seamless communication. A context-aware system will enhance the chatbot's ability to understand course materials, deadlines, and student interactions. The project aligns with SDG 4: Quality Education, improving student engagement and learning accessibility.

Voice-Enabled AI Chatbot with Context Awareness in LMS

Employability.Life

Neeraj Singh

Version Control:

Date of Creation	Name of Author	Version
20-02-2025	Mr. Manpreet Singh	Ver. 1.0

Confidentially:

Confidentiality	Intellectual Property
The document is meant to be used by the stake holders of Employability.Life and their customers. This document and any information contained herein are confidential and proprietary to Employability.Life ("Disclosing Party"). Recipient agrees to maintain the confidentiality of this information and to use it solely for the purpose of [purpose of sharing the information]. Recipient shall not disclose, distribute, or reproduce this information in any form without the prior written consent of the Disclosing Party.	The Intellectual Property of the document belongs to Employability.Life

Contents

1. Project Scope

1.1 Use Case Synopsis

Many students face difficulties navigating the LMS, retrieving relevant course materials, and staying on top of deadlines. This project introduces a voice-enabled chatbot that enhances student engagement by providing hands-free interaction with the LMS. The chatbot will understand course context, track student progress, and deliver personalized academic assistance.

1.2 Objective of the Project

- ✓ Develop a voice-based AI assistant that integrates with the LMS.
- ✓ Enable context-aware responses based on student progress and coursework.
- ✓ Provide real-time assistance on assignments, schedules, and deadlines.
- ✓ Improve learning accessibility through speech-to-text and text-to-speech capabilities.

1.3 Stakeholders

- ✓ **Students:** Primary users who will interact with the chatbot for academic assistance.
- ✓ **Instructors:** Can leverage chatbot insights to monitor student engagement.
- ✓ **LMS Administrators**: Responsible for chatbot integration and system maintenance.

1.4 Scope

- ✓ **Voice Interaction**: Implement NLP and speech recognition for natural conversation.
- ✓ **Context Awareness:** Integrate with LMS APIs to track assignments and deadlines.
- ✓ **Personalization:** Adapt chatbot responses based on individual student progress.
- ✓ **Multi-Modal Support:** Enable both voice and text-based interactions.

1.5 Activities To be Performed.

- ✓ **LMS Integration:** Connect chatbot to the institution's LMS.
- ✓ **NLP Training:** Train the chatbot on academic queries and course-related conversations.
- ✓ **Speech Processing:** Implement speech-to-text and text-to-speech functionalities.
- ✓ **User Interface:** Develop a dashboard for chatbot configuration and analytics.

1.6 Technologies Used

- ✓ **Programming Languages:** Python, JavaScript (Node.js)
- ✓ **AI & NLP:** OpenAI GPT, Google Dialogflow, or Rasa
- ✓ **Speech Recognition:** Google Speech-to-Text, Amazon Polly
- ✓ **LMS Integration:** Moodle, Blackboard, Canvas APIs

✓ **Frontend:** React, Vue.js for chatbot UI

2. Project Details

The project follows an agile approach, structuring development into three key phases.

3. Sprint Plan

3.1 Sprint 1 - LMS Integration & Data Collection

Objective: Establish LMS connectivity and enable chatbot access to course materials.

Epic 1: LMS Integration & Data Retrieval

User Story 1.1: As a developer, I want to connect the chatbot to the LMS API so that it can retrieve course-related data.

✅ Tasks:

- Identify LMS API endpoints for accessing assignments, discussions, and grades.
- Develop authentication mechanisms for secure API access.
- Store relevant student progress data in a structured format

User Story 1.2: As a data engineer, I want to preprocess course data so that the chatbot can understand context.

✅ Tasks:

- Clean and structure LMS data for chatbot training.
- Define common academic queries based on course materials.
- Implement a sample dataset for testing chatbot responses.

Sprint 1 Deliverables:

📌 **Deliverable 1:** API documentation for LMS integration.
📌 **Deliverable 2:** Structured dataset of academic queries.
📌 **Deliverable 3:** Data processing report for chatbot training.

3.2 Sprint 2 - Voice & NLP Model Development

Objective: Implement natural language understanding and speech recognition capabilities.

Epic 2: Implement Voice and AI Capabilities

User Story 2.1: As an AI developer, I want the chatbot to understand academic questions so that it can provide relevant answers.

✅ **Tasks**:

- Train the chatbot using NLP frameworks (Dialogflow, Rasa, GPT).
- Develop predefined intents for frequently asked questions (assignment deadlines, course explanations).
- Test model accuracy and improve response quality.

User Story 2.2: As a software engineer, I want to integrate speech-to-text functionality so that students can interact with the chatbot using voice commands.

✅ **Tasks**:

- Implement Google Speech-to-Text for voice input.
- Develop real-time voice recognition for hands-free interaction.
- Optimize latency for quick response times.

Sprint 2 Deliverables:

📌 **Deliverable 4:** NLP model trained on academic queries.
📌 **Deliverable 5:** Functional voice-enabled chatbot prototype.
📌 **Deliverable 6:** Model performance evaluation report.

3.3 Sprint 3 - UI, Dashboard & Deployment

Objective: Develop a frontend interface and finalize deployment.

Epic 3: Chatbot UI & Dashboard

User Story 3.1: As a student, I want a chatbot interface so that I can interact with it easily.

✅ **Tasks**:

- Develop a chatbot UI using React/Vue.js.
- Implement both voice and text interaction options.
- Ensure mobile-friendly design.

User Story 3.2: As an instructor, I want an analytics dashboard to monitor chatbot usage and student engagement.

✅ **Tasks**:

- Create a dashboard to visualize chatbot interactions.
- Implement logging and analytics to track student engagement.
- Generate automated reports on student queries and concerns.

Sprint 3 Deliverables:

📌 **Deliverable 7:** Chatbot UI prototype.
📌 **Deliverable 8:** Instructor dashboard with chatbot analytics.
📌 **Deliverable 9:** Final project report and user documentation.

4. Task Assignment Strategy

4.1 Skill Level Consideration

Tasks will be assigned based on students' strengths in AI, backend integration, and UI/UX development.

4.2 Career Goals Alignment

- Students interested in **AI/ML** can work on chatbot training.
- Those focused on **frontend development** can build the chatbot UI.
- **Cloud enthusiasts** can handle deployment and LMS integration.

4.3 Collaboration Opportunities

Encourage **pair programming** and **cross-functional team discussions** to enhance learning.

5. Sprint Deliverable

Sprint	Deliverables
Sprint 1	LMS API Integration Report, Pre-processed Course Data Report
Sprint 2	NLP Training Documentation, Speech-to-Text Functionality Report.
Sprint 3	Chatbot UI Wireframe & Design Document, Instructor Dashboard Report.

6. Deliverable Template Structure (Sprint Wise)

6.1 Sprint 1 Week 1 Deliverable

Deliverable 1: LMS API Integration Report

1. Introduction

- **Objective:** Establish a connection between the AI chatbot and the LMS for retrieving relevant course data.
- **Scope:** Integration with LMS APIs (Canvas, Moodle, Blackboard) to fetch assignments, grades, schedules.

2. LMS API Endpoints

API Endpoint	Data Retrieved	Method	Authentication
/courses/{id}/assignments	Assignments List	GET	OAuth Token
/users/{id}/grades	Student Grades	GET	OAuth Token

/courses/{id}/discussions	Course Discussions	GET	OAuth Token

3. Data Retrieval Process

- **API Authentication Example**:

  Python:

  import requests

  headers = {"Authorization": "Bearer YOUR_TOKEN"}

  response = requests.get("https://lms-api.com/courses/assignments", headers=headers)

  data = response.json()

- **Challenges & Solutions**
  - **Access Restrictions:** Worked with LMS admin to generate API keys.
  - **Data Inconsistencies:** Standardized API responses for uniformity.

4. Next Steps

- Implement data structuring and preprocessing in Week 2.

6.2 Sprint 1 Week 2 deliverable

Deliverable 2: LMS Data Preprocessing Report

1. Introduction

- **Objective:** Clean and structure LMS data for AI chatbot training.

1. Data Processing Steps

Field	Issue Identified	Cleaning Method
Assignment Deadlines	Missing values	Replaced with course average deadline
Grade Format	Inconsistent (letter/percent)	Standardized to percentage (0-100)
Discussion Responses	HTML tags present	Stripped HTML formatting

3. Normalization and Transformation

- **Before Normalization Example**

Csv:

Student_ID, Assignment, Grade, Due_Date

101, "Math Homework", A, NULL

102, "Science Project", 75%, "2024-05-10"

- **After Normalization**

Csv:

Student_ID, Assignment, Grade (%), Due_Date

101, "Math Homework", 90, "2024-05-15"

102, "Science Project", 75, "2024-05-10"

4. Challenges & Next Steps

- **Challenge:** Handling missing due dates.
- **Solution:** Imputed missing dates based on similar assignments.
- **Next Step:** Training the chatbot in Sprint 2

6.3 Sprint 2 Week 3 deliverable Template

Deliverable 3: NLP Model Training Report

1. Introduction

- **Objective:** Train a natural language processing model to understand student queries.

2. Defined Intents & Responses

Intent	Sample User Query	Response Example
Check Assignment	"What's due next week?"	"Your next assignment is Math Homework due May 10."
Grade Inquiry	"What's my grade in History?"	"Your current grade in History is 88%."
Course Material	"Send me lecture notes for Physics."	"Here is the latest Physics lecture note."

3. NLP Model Training Process

- **Preprocessing Steps**:

Python:

```
import spacy
```

```
nlp = spacy.load("en_core_web_sm")
text = "What's my grade in History?"
doc = nlp(text)
tokens = [token.text for token in doc]
```

- **Chatbot Model Training**:

Python:

```
from rasa.nlu.training_data import load_data
from rasa.nlu.model import Trainer
trainer = Trainer("config.yml")
trainer.train(load_data("training_data.json"))
```

4. Next Steps

- Implement voice recognition in Week 4.

6.4 Sprint 2 Week 4 deliverable Template

Deliverable 4: Speech-to-Text Integration Report

1. Introduction

- **Objective:** Implement speech-to-text functionality for hands-free student interaction.

2. Speech Recognition Implementation

Component	Technology Used
Speech-to-Text	Google Speech API, Whisper
Text-to-Speech	Amazon Polly, Azure Speech Services

- **Example Implementation**

Python:

```
import speech_recognition as sr
recognizer = sr.Recognizer()
with sr.Microphone() as source:
```

```
audio = recognizer.listen(source)
text = recognizer.recognize_google(audio)
```

3. Next Steps

- Implement chatbot UI in Sprint 3.

6.5 Sprint 3 Week 5 deliverable Template

Deliverable 5: Chatbot UI Design Document

1. Introduction

Objective: Design a user-friendly chatbot interface.

2. Wireframe

(Screenshot of chatbot interface)

3. Dashboard Features

Feature	Description
Voice & Text Chat	Users can type or speak queries
Context Awareness	Bot remembers previous interactions
Personalized Dashboard	Displays student progress and reminders

4. Next Steps

Develop chatbot UI in Week 6.

6.6 Sprint 3 Week 6 deliverable Template

Deliverable 6: Final AI Chatbot Report

Document Title: Final AI Chatbot Performance & Deployment Report
Version: 1.0
Author: [Student Name]
Date: [DD/MM/YYYY]

1. Introduction

- **Objective:** Summarize chatbot performance and deployment.

2. Key Findings

Test Case	Expected Output	Actual Output	Status
"What's due next week?"	"Math Homework due May 10."	"Math Homework due May 10."	✅ Passed
"Read my grade in Physics"	"Your grade in Physics is 85%."	"Your grade in Physics is 85%."	✅ Passed

3. Deployment Strategy

- **Cloud Deployment:** AWS Lambda & API Gateway.
- **Database:** Firebase for real-time storage.

4. Future Enhancements

- Improve multi-language support.
- Add adaptive learning recommendations.

Final Submission

🎯 **Live demo** (voice & text interaction)
🎯 **Documentation** (all deliverables compiled)
🎯 **Show & Tell Video** (chatbot capabilities & learnings)