Dataset Design:

Column Name	Description	Example	Type
Customer_ID	Unique ID for each applicant	C001	String
Name	Applicant's full name	Nishtha Sharma	String
Age	Applicant's age	27	Integer
Gender	Gender	F	Categorical
Marital_Status	Married/Single	Single	Categorical
Employment_Type	Job / Self-employed / Business	Job	Categorical
Monthly_Income	Declared income in INR	80000	Numeric
Loan_Amount_Requested	Loan amount in INR	500000	Numeric
Credit_Score	Credit score from 300–900	750	Numeric
Existing_EMI	Total EMI paid per month	12000	Numeric
Loan_Tenure_Years	Tenure requested	5	Numeric
Dependents	Number of dependents	2	Numeric
Bank_Statement_Verified	Whether document verified	Yes / No	Categorical
Voice_Verified	Whether voice authentication passed	Yes / No	Categorical
Document_Verified	Whether uploaded documents are verified	Yes / No	Categorical
Debt_to_Income_Ratio	EMI / Monthly Income	0.15	Derived (Numeric)
Total_Liabilities	Outstanding debt	100000	Numeric
Past_Defaults	Previous loan defaults count	0	Numeric
Region	Customer location	Delhi	Categorical
Loan_Purpose	Reason for loan	Home / Education / Car	Categorical
Eligibility_Score	Model predicted probability (0–1)	0.84	Numeric
Loan_Approved	Target label: Approved(1) / Denied(0)	1	Target

TWO ACCESS CHANNELS — SAME INTELLIGENCE

Mode	Input Type	Interface	AI Brain	Output
Chatbot	Text	React Web App	Bedrock Agent	Text reply
Voice Agent	Voice (call)	Amazon Connect	Bedrock Agent + Polly	Spoken reply

---

🧩 **How the Chatbot Works (Web Mode)**

🧠 **1. User opens your website (React app)**

- Sees a chatbot window (like ChatGPT-style chat).
- Can type or use mic (optional speech-to-text).

💬 **2. Chat message sent to backend (FastAPI)**

- FastAPI forwards message to **Amazon Bedrock** (LLM).

🗨️ **3. Bedrock Agent replies**

- If it needs document verification:
  - Generates a pre-signed S3 upload link.
  - Sends it to the user in chat.
- If enough data collected:
  - Calls the ML model endpoint on **SageMaker**.
  - Returns eligibility score.

📄 **4. Agent explains result**

"You are 86% eligible for the loan. Would you like to proceed with manager verification?"

✅ **5. Final approval**

- Data automatically synced to **Bank Manager Dashboard** for review.

☎️ **How the Voice Agent Works (Call Mode)**

Same logic, but via **Amazon Connect**:

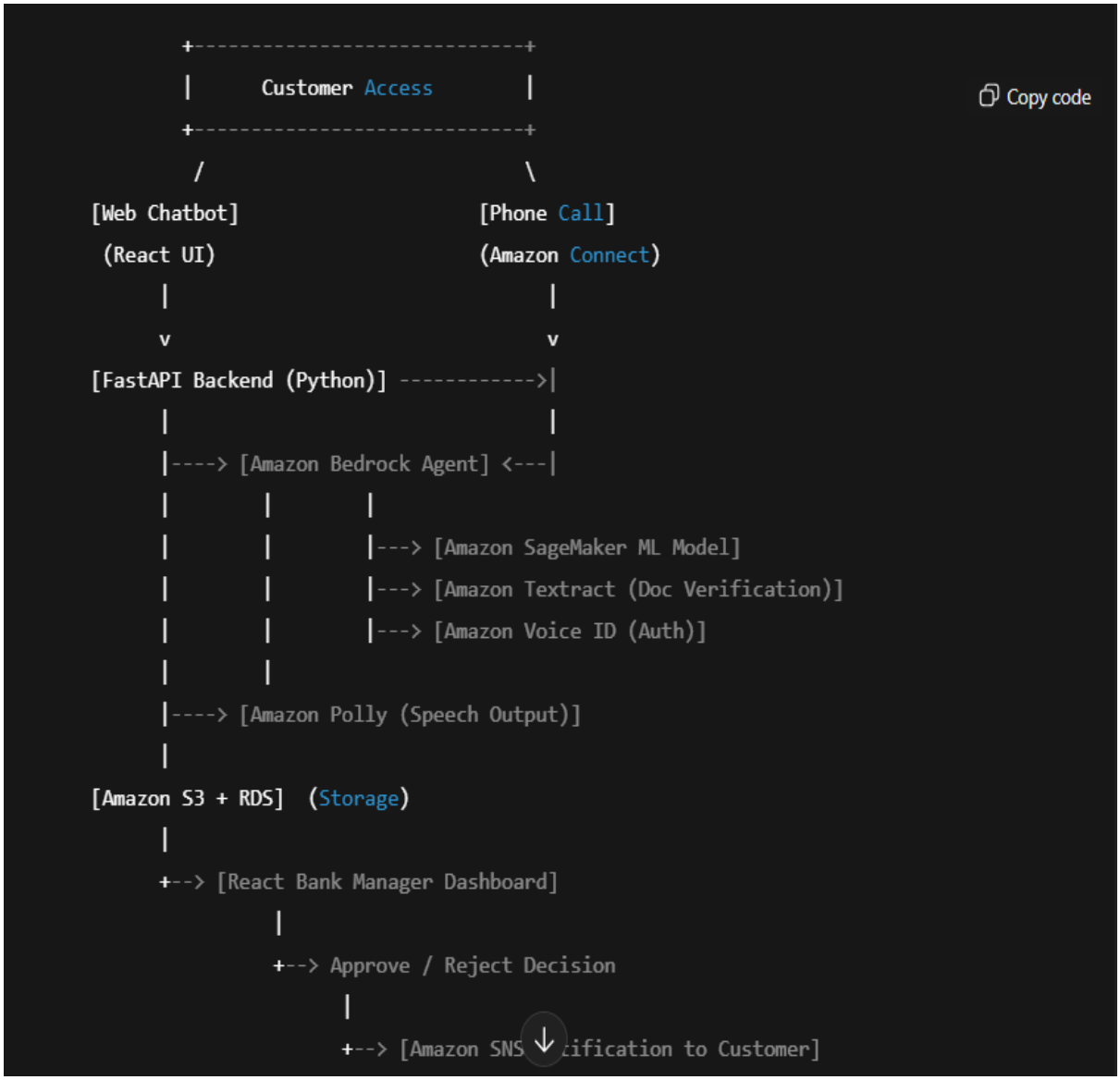
1. Caller speaks → **Transcribe** converts to text
2. Text → **Bedrock** → reasoning + response
3. Response → **Polly** → spoken voice → back to caller
4. All results stored in backend + sent to manager dashboard

🔗 **Common Intelligence Core (Shared AI Logic)**

Both chatbot and voice agent share the same backend intelligence:

- Same **FastAPI backend**
- Same **Amazon Bedrock agent**
- Same **SageMaker model** for predictions
- Same **Textract** for document verification
- Same **Voice ID** for speaker authentication
- Same **RDS/DynamoDB** for data storage

🧰 Simplified Technical Flow



🎯 Project Flow in Simple Steps (Chat + Voice Unified)

🧩 Step 1: User starts

- Either calls the AI agent 📞 or opens the chat 💬 on the website.

🧠 Step 2: AI conversation

- AI collects name, income, loan amount, and credit score.
- If on call → voice converted to text.
- If on chat → text directly received.

🔐 Step 3: Authentication

- Voice-based verification (for calls)
- OTP/email verification (for chat users)

📄 Step 4: Document Verification

- System generates a pre-signed S3 upload link.
- User uploads document → Textract reads and verifies it.

🤖 Step 5: Loan Eligibility Prediction

- Backend sends all info to SageMaker model → gets eligibility score.

🗨️ Step 6: AI explains result

- For call: spoken explanation (via Polly)
- For chat: text explanation in web chatbot

🕴️ Step 7: Manager Review

- Bank manager checks details on React dashboard → Approves/Rejects.

📲 Step 8: Notification & Report

- Customer receives SMS/email with final result.
- Report saved in S3.

📜 Technologies in Use

Function	Tool / Service	Description
Chat Interface	React	Customer text-based chatbot
Voice Call	Amazon Connect	AI call system
Speech-to-Text	Amazon Transcribe	Converts spoken words to text

Function	Tool / Service	Description
AI Reasoning	Amazon Bedrock	Chat + voice conversation logic
Text-to-Speech	Amazon Polly	Converts text replies to speech
Voice Authentication	Amazon Voice ID	Identifies speaker
Document Verification	Amazon Textract	Extracts data from uploads
Eligibility Prediction	Amazon SageMaker	Predicts loan approval score
Backend	FastAPI (Python)	Connects all components
Manager Dashboard	React	For human approval
Notifications	Amazon SNS	SMS/email to customer
Data Storage	S3 + RDS/DynamoDB	Documents + metadata
Monitoring	CloudWatch	Tracks logs & performance

Goal

To build an intelligent system that can **talk to customers over a phone call or chat**,
collect their information, **verify their identity and documents**,
predict loan eligibility, and let the **bank manager manually approve or reject** the application.

🚀 **How It Works — In Simple Words**

Let's imagine a customer, "Rahul", calling your AI Loan Assistant.

Step 1: User Connects

Rahul dials your **bank's helpline number** (hosted on **Amazon Connect**).

- The call is picked up by your **AI assistant** — not a human.
- The assistant greets Rahul and starts the conversation.

🛠️ *Service Used:*
Amazon Connect → Handles phone call connection.

Step 2: Voice to Text Conversion

Rahul speaks, and his voice is converted into text instantly.

🛠️ *Service Used:*
Amazon Transcribe (Streaming) → Converts live speech into text.

Example:

“My name is Rahul Sharma, I want to apply for a loan of ₹5,00,000.”

Step 3: AI Agent Understands and Responds

The text is sent to your **AI Agent (Amazon Bedrock)**, which understands what Rahul wants.

- The agent replies using human-like speech generated through **Amazon Polly**.
- Rahul can actually *talk* with the agent in real time.

🛠️ *Services Used:*
Amazon Bedrock → Brain of the system (understands conversation).
Amazon Polly → Converts AI text responses back to natural voice.

Step 4: Voice Verification (Authentication)

To ensure Rahul is genuine, the agent asks:

“Please say, ‘My name is Rahul Sharma’ for verification.”

- The system records this and checks if Rahul’s voice matches his previous record.
- If not available, it stores this as his new voiceprint.

🛠️ *Service Used:*
Amazon Voice ID / Rekognition → Compares the speaker’s voice for authentication.

Step 5: Collect Loan Details

The agent asks simple questions like:

- What is your monthly income?
- What is your credit score?
- Do you have any existing loans?

The AI listens, understands, and stores answers.

🛠️ *Handled by:*
Amazon Bedrock (Agent) + **FastAPI backend (Python)**.

Step 6: Request and Verify Documents

If needed, the AI says:

“Please upload your latest 3-month bank statement for verification.”

- The backend creates a **secure upload link** (S3 Pre-signed URL).

- The link is sent to Rahul via **SMS or email**.
- Rahul uploads his document.

Once uploaded:

- A **Lambda function** runs automatically.
- It sends the document to **Amazon Textract**, which extracts text (like salary, account number, transactions).
- The system checks if the salary mentioned by Rahul matches the document.

🛠️ *Services Used:*
Amazon S3 → Stores documents.
AWS Lambda → Triggers document analysis.
Amazon Textract → Extracts text and numbers from uploaded documents.
Amazon SNS → Sends upload links and alerts via SMS/email.

Step 7: Loan Eligibility Prediction

Now that we have:

- Income
- Credit Score
- Loan Amount
- Voice Verified ✅
- Documents Verified ✅

All these details are sent to your **Machine Learning Model** hosted on **Amazon SageMaker**.

- The model checks if Rahul is eligible for a loan or not.
- It outputs a score like 0.86 → meaning 86% chance of approval.

🛠️ *Service Used:*
Amazon SageMaker → Runs trained ML model (e.g., XGBoost) for loan eligibility prediction.
SHAP Explainability → Explains why the decision was made (e.g., "Credit score and income helped the result").

Step 8: AI Agent Communicates the Result

The AI agent then explains the decision politely:

"Based on your verified income and credit score, you are 86% eligible for a ₹5,00,000 loan."

If Rahul's eligibility is low, the agent can also give suggestions:

"To improve your chances, you can reduce existing EMIs or increase your credit score."

🛠️ *Service Used:*
Amazon Bedrock + Polly → Conversational reasoning and speech generation.

Step 9: Manager Review (Manual Approval)

After the AI prediction, details go to a **Bank Manager Dashboard**.

The manager can:

- See customer info
- Check AI result
- Review documents
- Approve or Reject loan manually

🛠️ *Tools Used:*
React (Frontend) → Dashboard UI
FastAPI (Backend) → Connects with AWS and database
Amazon RDS / DynamoDB → Stores decisions and user data

Step 10: Notify Customer + Generate Report

Once the manager approves/rejects:

- The system sends Rahul an **SMS or email notification**.
- A **PDF report** is generated (with eligibility score, verification results, and manager decision).
- Report is stored in **S3**.

🛠️ *Service Used:*
Amazon SNS → Sends result notifications
Amazon S3 → Stores final PDF report

Step 11: Monitoring and Security

Everything (calls, model predictions, uploads) is monitored automatically.

🛠️ *Service Used:*
Amazon CloudWatch → Logs all activities and helps debug issues.
AWS IAM + KMS → Ensures data security and encryption.

🧰 **Technology Summary**

Layer	Tools / Services Used	Role
Frontend (User & Manager)	React	Chat UI + Manager dashboard

Layer	Tools / Services Used	Role
Backend	FastAPI (Python)	Connects AI, ML, and AWS services
Voice & Call Handling	Amazon Connect	Real phone calls
Speech Recognition	Amazon Transcribe	Converts speech to text
AI Conversation	Amazon Bedrock	Conversation & reasoning
Voice Output	Amazon Polly	AI text → speech
Voice Verification	Amazon Voice ID	Authenticates caller
ML Prediction	Amazon SageMaker (XGBoost)	Predicts eligibility
Document Processing	Amazon Textract + S3 + Lambda	Reads and verifies documents
Notifications	Amazon SNS	Sends messages & results
Database	DynamoDB / RDS	Stores user and manager data
Monitoring	Amazon CloudWatch	Logs and tracking

🧭 **Complete Flow (Step-by-Step Summary)**

1️⃣ Customer calls bank → Amazon Connect picks up

2️⃣ Connect → Transcribe (Speech → Text)

3️⃣ Text → FastAPI backend → Bedrock (Conversation)

4️⃣ Bedrock → Polly (Reply as Voice)

5️⃣ Voice verification → Amazon Voice ID

6️⃣ Document upload → S3 → Lambda → Textract

7️⃣ Data → SageMaker model → eligibility score

8️⃣ Bedrock explains result to customer

9️⃣ Bank manager reviews via React dashboard (Approve/Reject)

🔟 SNS notifies customer + PDF report saved in S3

🧩 **FINAL FLOWCHART — VOICE CALL–BASED AI SYSTEM**

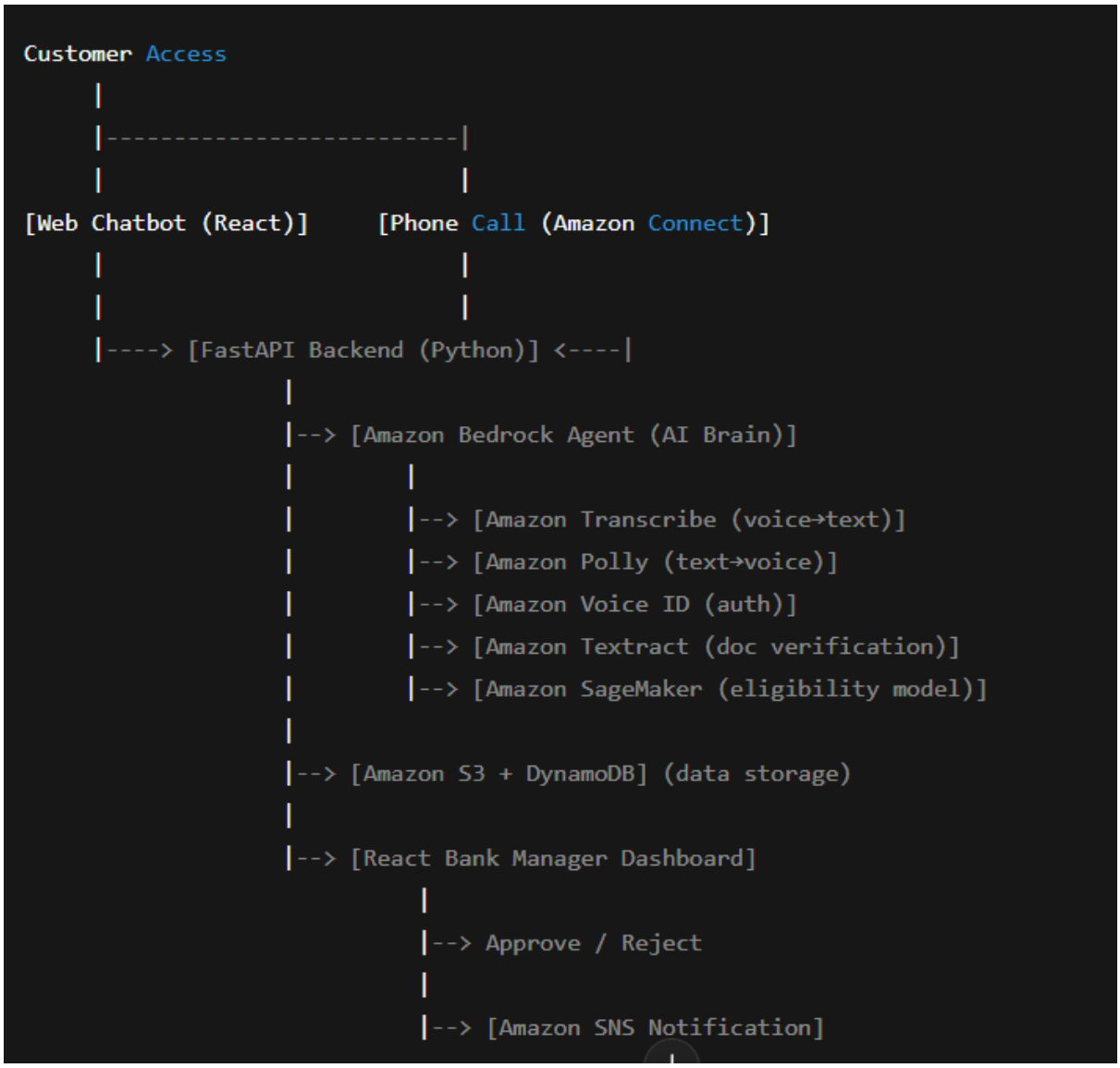
Here's the full visual representation of your system:

📊 **Final Flowchart Image**

✅ **This covers everything:**

- Real phone call with AI voice
- AI agent reasoning (Bedrock)
- Voice authentication (Voice ID)
- Document verification (Textract)
- Loan eligibility model (SageMaker)
- Manager review system (React + FastAPI)
- Customer notification + reports
- Monitoring and security (CloudWatch + IAM)

Combined (Chatbot + Voice AI Integration)



Here's the **complete tech stack** used in your project — divided by category so you can list it clearly in your documentation or report 👇

🧱 1. Frontend (User Interface)

Component	Technology	Purpose
Chatbot Interface	React.js	Web chat interface for users to interact via text
Voice Call Interface	Amazon Connect	Handles real phone calls with the AI agent
Manager Dashboard	React.js	Bank manager panel for approval/rejection
Styling/UI Library	Tailwind CSS / Material UI	Modern and responsive UI design

---

⚙️ 2. Backend (Application Logic)

Component	Technology	Purpose
Web Framework	Python + FastAPI	Main backend server connecting frontend and AWS services
API Gateway	AWS API Gateway	Securely exposes backend endpoints
Serverless Functions	AWS Lambda	Handles document verification, event triggers, and automation

---

🤖 3. AI & Machine Learning

Component	Service / Tool	Purpose
AI Agent (LLM)	Amazon Bedrock (Claude / Titan Model)	Conversational reasoning and intelligent dialogue
Speech Recognition	Amazon Transcribe (Streaming)	Converts user speech → text during calls
Text-to-Speech	Amazon Polly	Converts AI replies → speech
Speaker Verification	Amazon Voice ID	Authenticates caller's identity using voice
Document Analysis	Amazon Textract	Extracts structured data from uploaded documents
ML Prediction Model	Amazon SageMaker (XGBoost / LightGBM)	Predicts loan eligibility score
Explainability	SHAP (on SageMaker / Python)	Explains model decisions in plain language

---

☁️ 4. Cloud Infrastructure & Storage

Component	Service / Tool	Purpose
Cloud Platform	AWS	Main cloud environment
Storage	Amazon S3	Stores documents, reports, and model files
Database	Amazon DynamoDB / Amazon RDS	Saves user data, predictions, and manager decisions
Authentication &	AWS IAM + AWS KMS	Role-based access and encryption

Component	Service / Tool	Purpose
Security		
Monitoring	Amazon CloudWatch	Logs and monitors all activities

---

📡 5. Communication & Notifications

Component	Service / Tool	Purpose
Messaging	Amazon SNS (Simple Notification Service)	Sends SMS or email notifications to users
Email (Optional)	AWS SES	For sending email confirmations or OTPs

---

🧰 6. Development & Integration Tools

Tool	Purpose
VS Code / PyCharm	Code editor for backend and frontend
Git + GitHub / GitLab	Version control
Postman	API testing
Docker (Optional)	Containerization for backend deployment
Render / Netlify / Hugging Face Spaces (Optional)	Web hosting alternatives for demo
AWS Console	Cloud resource management

💾 7. Dataset & Model Training

Component	Tool	Purpose
Dataset Creation	Excel / Google Sheets / Python Faker	Generate dummy loan applicant data
Model Training Notebook	Amazon SageMaker Notebook / Jupyter	Train ML model using CSV dataset
Data Visualization	Matplotlib / Seaborn (Python)	Explore and plot data insights

---

🔒 8. Security & Compliance

Component	Tool / Service	Purpose

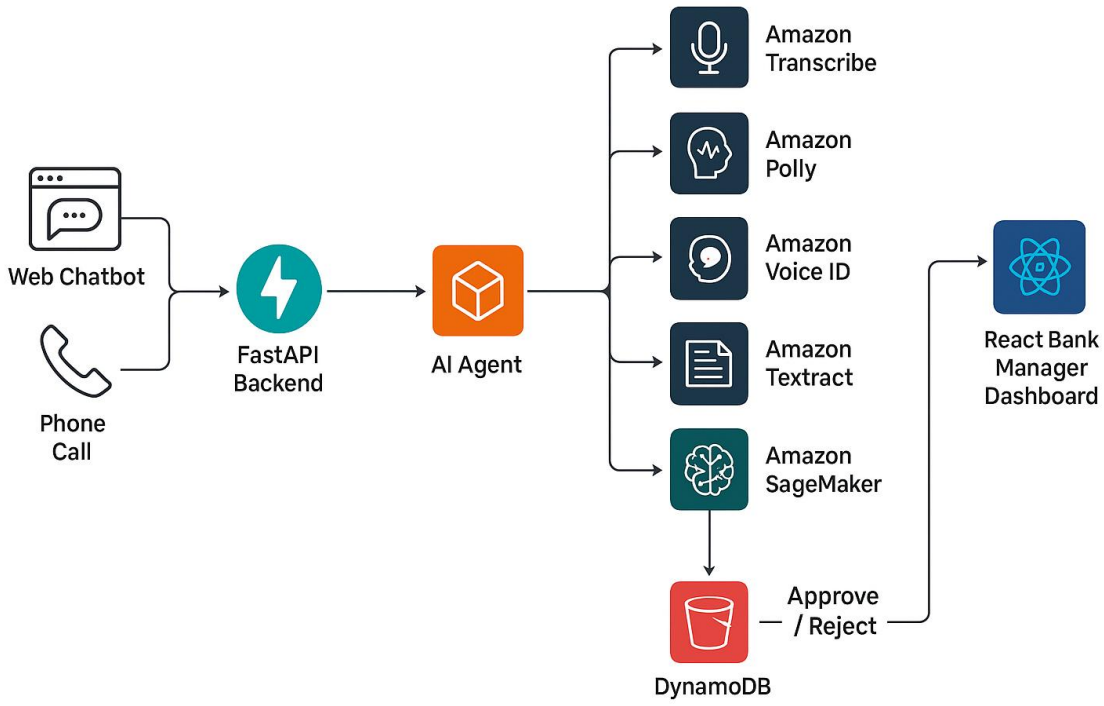
Component	Tool / Service	Purpose
Encryption	AWS KMS	Encrypt sensitive data in S3 & RDS
Access Control	AWS IAM	Role-based service permissions
Data Privacy	GDPR-Compliant Design	Ensures safe handling of user data

---

✅ **In short (Key Stack Overview)**

Frontend: React.js, Tailwind CSS
Backend: Python (FastAPI), AWS Lambda
AI & ML: Amazon Bedrock, SageMaker, Textract, Polly, Transcribe, Voice ID
Database: DynamoDB / RDS
Storage: Amazon S3
Notification: Amazon SNS
Authentication: Voice ID + IAM
Dev Tools: VS Code, GitHub, Postman

Would you like me to make this into a **visual "Tech Stack Diagram" (with logos/icons)** — for your final PPT or report? It'll show each layer (Frontend, Backend, AI, Cloud) with their respective tools neatly organized.



AI-Powered Voice & Chat Loan Eligibility System