User Acceptance Testing (UAT) Template

Date	18 February 2026
Team ID	LTVIP2026TMIDS66060
Project Name	IntelliSQL: Intelligent SQL Querying with LLMs Using Gemini Pro
Maximum Marks	

Project Overview:

Project Name **:** IntelliSQL: Intelligent SQL Querying with LLMs Using Gemini Pro

Project Description **:** A Streamlit-based web application that leverages Google's Gemini 1.5 Flash model to translate natural language English into executable SQL queries, allowing non-technical users to retrieve student data from a SQLite database.

Project Version **:** v1.0.0

Testing Period **:** 18-February-2026 to 19-February-2026

Testing Scope:

Features & Functionalities to be Tested

- **Secure API Key Loading**: Verifying the system correctly retrieves the GOOGLE_API_KEY from the .env file without exposure.
- **Natural Language Translation**: Testing the accuracy of the Gemini Flash model in converting English questions into valid SQL.
- **Regex SQL Extraction**: Ensuring the logic correctly strips conversational AI text to isolate raw SQL code.
- **Database Query Execution**: Confirming that the read_query function successfully fetches data from the STUDENTS table.
- **UI Rendering**: Testing the Streamlit interface for table display and sidebar navigation (Home, About, Query Tool).

User Stories / Requirements

- **Non-Technical User**: As a user, I can ask questions in plain English (e.g., "Show me students in Data Science") and see a table of results.
- **Data Accuracy**: As a user, I can view specific student marks and placement companies fetched directly from the database.
- **System Admin**: As an admin, I can securely manage API configurations and local database records.

Testing Environment:

- **URL/Location**: http://localhost:8501 (Default Streamlit port) or [Deployed URL on Streamlit Cloud]
- **Database**: Local data.db (SQLite) containing student records.
- **Prerequisites**: Python 3.x environment

  GOOGLE_API_KEY configured in .env file

Test Cases:

Test Case ID	Test Scenario	Test Steps	Expected Result	Actual Result	Pass/Fail
TC-001	API Authentication	1. Ensure .env file contains valid key. 2. Launch Streamlit app.	App initializes without "API Key not found" errors.	Correctly loaded	Pass
TC-002	Basic Query Translation	1. Enter "Show all students". 2. Click "Get Answer".	System generates `SELECT * FROM STUDENTS;`.	Exact match	Pass
TC-003	Filtered Query Accuracy	1. Enter "Who is in Data Science?". 2. Click "Get Answer".	SQL includes `WHERE CLASS='Data Science'`.	Correct filter	Pass
TC-004	Regex Sanitization	1. Trigger AI response with markdown code blocks.	System extracts only the SQL string between tags.	Clean SQL	Pass
TC-005	Database Execution	1. Submit a valid English question. 2. Wait for processing.	Results from `data.db` are fetched and displayed.	Data retrieved	Pass
TC-006	UI Data Rendering	1. Observe the output area after a query.	Data is displayed in a structured, readable table.	Proper table	Pass

TC-007	Sidebar Navigation	1. Click "About" in the sidebar. 2. Click "Home".	The main page content updates correctly for each section.	Smooth navigation	Pass
TC-008	Negative Scenario (Invalid Input)	1. Enter random gibberish (e.g., "xyzabc"). 2. Click "Get Answer".	System displays a friendly error or handles the empty result safely.	Handled safely	Pass
TC-009	Numerical Aggregation	1. Enter "What is the average marks?".	System generates `SELECT AVG(MARKS) FROM STUDENTS;`.	Accurate SQL	Pass
TC-010	Specific Record Search	1. Enter "Find student named Sijo".	SQL targets the `NAME` column correctly.	Found record	Pass

Bug Tracking:

Bug ID	Bug Description	Steps to reproduce	Severity	Status	Additional feedback
BG-001	AI response includes conversational text in SQL execution	1. Enter a vague query. 2. Submit.	High	Closed	Fixed by implementing Regex extraction to isolate raw SQL.
BG-002	Sidebar navigation doesn't reset query results	1. Run a query. 2. Switch to 'About' page.	**Medium**	**Open**	Results persist in memory; need to clear session state on page change.
BG-003	Empty database error on first launch	1. Delete data.db. 2. Run `app.py`.	**High**	**Closed**	Resolved by ensuring `sql.py` is executed to seed the database before app launch.

BG-004	Markdown formatting (```sql) causing execution failure	1. Input query. 2. Receive LLM response with tags.	**High**	**Closed**	Regex pattern updated to strip markdown formatting before sending to SQLite.
BG-005	UI text overflow on mobile screens	1. Open app on mobile. 2. View wide data table.	**Low**	**In Progress**	Need to adjust CSS for responsive table scrolling.
BG-006	Invalid SQL syntax for complex phrasing	1. Enter slang or very long query.	**Medium**	**Open**	Model occasionally hallucinates non-existent columns; requires prompt tuning.

Sign-off:

Tester Name: Shaik Ziaur Rahaman

Date: 18 February 2026

Signature: Shaik Ziaur Rahaman

Notes:

- All test cases include positive and negative paths.
- Any failed test cases will be retested after resolution.
- All critical bugs must be resolved before production deployment.
- Product Owner and Project Manager approval is required before go-live.