Human Evaluation on DALLE

This form is intended to evaluate the quality of the results produced by DALLE, an LLM-based assistant designed to extract microservice-based architecture designs from textual input.

Overview of the Evaluation

You will be asked to evaluate *4 different projects*. Each project includes:

- A *brief description* of the system to be generated
- A set of *user stories* related to the system

These inputs are provided to the DALLE system, which then produces an architectural design. Among others outputs, DALLE identifies:

- The **list of microservices**
- The **microservice and communication patterns** (based on the reference book)

Evaluation Process

For each project, you will be presented with:

- The original *system description* and *user stories*;
- A visual representation of the architecture generated by DALLE, including microservice patterns.

You are asked to evaluate whether the identified **data and communication style patterns** are appropriate and consistent with the project requirements.

Each evaluation question uses a **Likert scale from 1 to 7**, where:

- 1 = Strongly Disagree
- 7 = Strongly Agree

You may also provide **additional comments or observations** for each project in the dedicated Observation section.

Reference Patterns

The evaluation is based on the following architecture design patterns (as defined in the reference book):
Communication Style Patterns:

- Shared Database
- Database per Service

Data Style Patterns:

- API Composition
- CQRS
- Saga
- Aggregate
- Event Sourcing
- Domain Event

Final Questions

At the end of the form, you will find a few general questions about the perceived utility and effectiveness of DALLE as a tool for architectural design support.

* Indicates required question

1. Rate your experience in designing systems based on microservices architecture: *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

2. How many years of experience do you have? *

3. What is your age? *

4. In what gender do you identify yourself? *

Mark only one oval.

- Male
- Female
- Prefer not to say

5. Have you ever evaluated systems' architectures? *

Mark only one oval.

- ◯ Yes
- ◯ No

6. Describe your role in the SE field *

Mark only one oval.

- ◯ Practitioner
- ◯ Academic

ExamRush

SYSTEM DESCRIPTION:
ExamRush is a mobile application designed to help students study interactively through multiple-choice quizzes.
The application allows users to create a profile, choose a question deck, and answer questions using both touch and the phone's motion sensor.
The application supports multi-user functionality, an interactive and animated UI, and utilizes the phone's accelerometer for navigation between questions.

USER STORIES:
1) As a new user, I want to register an account using my email and password so that I can access the application.
2) As a user, I want to log in using my credentials so that I can resume using the application.
3) As a user, I want to upload and edit my profile picture so that I can personalize my account.
4) As a user, I want to apply a cartoon filter to my profile picture so that it looks fun and unique.
5) As a user, I want an interactive and animated user interface so that the application is engaging and easy to use.
6) As a student, I want to browse and select from multiple question decks so that I can choose a topic to study.
7) As a student, I want to answer multiple-choice questions interactively so that I can test my knowledge.
8) As a student, I want to search for decks using a search bar so that I can quickly find decks by topic, subject, or keyword.
9) As a student, I want to see statistics about my performance (e.g., average score, strengths, weaknesses, progress over time) so that I can track my knowledge and identify areas for improvement.
10) As a teacher, I want to create and upload multiple-choice question decks so that my students can use them for studying.
11) As a teacher, I want to test the decks I upload by answering the questions myself so that I can ensure they are accurate and functional.
12)As a registered user, I want to be able to see my favorite list, so that I can check my current interests in recipes

Microservices

name	description
User Management Service	Handles user registration, login, profile management including uploading and editing profile pictures.
Image Processing Service	Applies image filters such as cartoon effects to user profile pictures.
UI/UX Service	Manages the interactive and animated user interface components of the application.
Question Deck Service	Manages creation, storage, browsing, searching, and selection of multiple-choice question decks.
Quiz Service	Handles the interactive answering of multiple-choice questions and supports navigation using touch and motion sensors.
Performance Analytics Service	Tracks and reports user performance statistics such as average scores, strengths, weaknesses, and progress over time.

Patterns

group_name	implementation_pattern	involved_microservices	explaination
Data Management	Database per Service	User Management Service,Image Processing Service,UI/UX Service,Question Deck Service,Quiz Service,Performance Analytics Service	Each microservice should own its own database to ensure loose coupling and independent scalability.
Data Consistency	Saga Pattern	User Management Service,Image Processing Service	To maintain data consistency across services that may need to coordinate transactions.
Client Experience	API Composition Pattern	UI/UX Service	To compose data from multiple services into a single response, improving client experience.

7 point likert scale

1. Strongly disagree
2. Disagree
3. Somehow disagree
4. Neither agree nor disagree
5. Somehow agree
6. Agree
7. Strongly agree

7. Is the division in microservice correct with respect to the context and the user stories? *

1 2 3 4 5 6 7

☆ ☆ ☆ ☆ ☆ ☆ ☆

8. Is **Saga Pattern** correctly applied? *

1 2 3 4 5 6 7

☆ ☆ ☆ ☆ ☆ ☆ ☆

9. Is **Database per Service Pattern** correctly applied? *

1 2 3 4 5 6 7

☆ ☆ ☆ ☆ ☆ ☆ ☆

10. Is **API Composition Pattern** correctly applied? *

1	2	3	4	5	6	7
☆	☆	☆	☆	☆	☆	☆

11. Observations

4-by-4

SYSTEM DESCRIPTION:
4-by-4 is an online platform that allows users to play the classic adversarial (m, n, k) games with gravity in a digital environment. The website features a standard login system, enabling users to create accounts, log in, and track their game statistics. Players can customize their gaming experience by adjusting board sizes and implementing chess-like timing settings to add a competitive edge. The platform is designed to provide a seamless and engaging experience for enthusiasts of all skill levels.

USER STORIES:
1. As a Connect Four fan, I want to play Connect Four online against other users so that I can enjoy the game more or less competitively.
2. As a player, I want to be able to register to the site so that I can customize my username.
3. As a player, I want to be able to log in the site so that I can access the same account every time.
4. As a user, I want to my credentials to be remembered so that I can access the site without typing them every time.
5. As a user, I want to be able to logout, so that other people on the same computer can't access my account.
6. As a user, I want to have helpful navigation buttons on all pages, so that it's easy to find my way around the site.
7. As a user, I want to look at my own profile, so that I can see details about my account.
8. As a user, I want to be able to change my username, so that I am not bound to a single name option forever.
9. As a user, I want to be able to change my password, so that I can be sure it is secure.
10. As a casual player, I want to be able to look at my aggregate statistics, so that I can estimate my skills and track my performance over time.
11. As a competitive player, I want to look at the winners of my previous games, so that I can see if there are common patterns between losses/wins.
12. As a competitive player, I want to look at replays of my previous matches, so that I can improve my gameplay.
13. As a player, I want to look at the settings (dimension and timing) of previous games, so that I can easily filter novel games.
14. As a competitive player, I want to look at the replays of other player's previous games, so that I can learn study their gameplay.
15. As a competitive player, I want to be able to know who created previous matches, to learn patterns in the games used.
16. As a player, I want to be able to look at active challenges, so that I can see if there's any open match I can join.
17. As a casual player, I want to be able to see who created a challenge, so that I can choose to play only with people I know.
18. As a competitive player, I want to look at a challenge creator's profile, so that I can check out whether they are a good player.
19. As a player, I want to be able to set varying board sizes when creating the challenge, to have a more novel experience
20. As a competitive player, I want to set chess-like timing settings (e.g., blitz, rapid, or custom time limits) so that I can challenge myself and others under time pressure.

21. As a player, I want all of the game logic to be handled automatically and fairly so that it isn't possible to cheat.
22. As a player, I want to be able to see whose turn it is, so that I am not waiting aimlessly.
23. As a player, I want to be able to chat with my opponent, so that I can have a conversation with them about the game.
24. As a competitive player, I want to view how much time me or my opponent have left, so that I can manage my time-per-move more effectively.
25. As a player, I want to click on the grid, so that I can place a piece during my turn.
26. As a player, I want to have the option to concede, so that I can end a losing gaming without having to wait.
27. As a player, I want to be able to offer a and accept a draw, so that I end a drawing game without having to wait.
28. As a competitive player, I want to be able to retire a draw offer if the opponent doesn't accept it, so that I can still try to win the game if they miss-play.
29. As a player, I want to be able to deny a draw offer, so that I can go for a win instead of settling for a draw.
30. As a player, I want to be able to immediately look at the match replay once it ends, so that I can review what happened.
31. As a beginner player, I want to be able to easily read who won the game and how it ended, so that I have a clear situation of whether I have won or not and how.
32. As a avid player, I want to have a button to exit the game once it ends, so that I can quickly start another one.
33. As a player, I want to have a button to go back to the profile from a replay, so that I am not forced to go through the entire replay.
34. As a player, I want to know how many moves there were in a previous match, so that I know how long it's going to take.
35. As a player, I want to be able to go through the replay move-by-move, so that I can see what happened gradually.
36. As a competitive Player, I want to be able to go back to the previous move in the replay, so that I can better analyse what happened more carefully.
37. As a beginner player, I want to be able to easily read who won the game and how it ended, so that I have a clear situation of who won and how.

Microservices

name	description
User Management Service	Handles user registration, login, logout, profile management, and credential storage.
Game Management Service	Manages game creation, game logic, board settings, timing settings, and game state including turns and moves.
Challenge Service	Manages active challenges, challenge creation, joining challenges, and challenge-related metadata such as creator information.
Statistics and History Service	Tracks and provides access to player statistics, game winners, previous game settings, and match metadata.
Replay Service	Handles storage, retrieval, and navigation of game replays including move-by-move playback and replay metadata.
Game Interaction Service	Manages in-game interactions such as chat, draw offers, conceding, and draw offer responses.
User Interface Navigation Service	Provides navigation aids and UI elements for easy movement across the platform.
Timing and Clock Service	Manages chess-like timing settings and displays remaining time for players during games.

Patterns

group_name	implementation_pattern	involved_microservices	explaination
Data Management Pattern	Database per Service Pattern	User Management Service,Game Management Service,Challenge Service,Statistics and History Service,Replay Service,Game Interaction Service,User Interface Navigation Service,Timing and Clock Service	Each microservice should have its own private database to ensure loose coupling and independent scalability.
Data Consistency Pattern	Saga Pattern	Game Management Service,Challenge Service,Statistics and History Service	To maintain data consistency across distributed transactions like game creation, challenge joining, and updating statistics.
Data Query Pattern	API Composition and CQRS Patterns	Statistics and History Service,Replay Service	For aggregating user statistics, game history, and replay data, use API Composition or CQRS for optimized querying.

7 point likert scale

1. Strongly disagree
2. Disagree
3. Somehow disagree
4. Neither agree nor disagree
5. Somehow agree
6. Agree
7. Strongly agree

12. Is the division in microservice correct with respect to the context and the user stories? *

1	2	3	4	5	6	7
☆	☆	☆	☆	☆	☆	☆

13. Is **Saga Pattern** correctly applied? *

1	2	3	4	5	6	7
☆	☆	☆	☆	☆	☆	☆

14. Is **CQRS Pattern** correctly applied? *

1	2	3	4	5	6	7
☆	☆	☆	☆	☆	☆	☆

15. Is **Database per Service Pattern** correctly applied? *

1	2	3	4	5	6	7
☆	☆	☆	☆	☆	☆	☆

16. Is **API Composition Pattern** correctly applied? *

1	2	3	4	5	6	7
☆	☆	☆	☆	☆	☆	☆

17. Observations

chronopic

SYSTEM DESCRIPTION:
ChronoPic is a photo management platform where users can upload images and automatically tag them based on age detection and optionally known persons. The system provides asynchronous processing using computer vision (face & age detection), and supports secure user authentication. Users can view, filter, and manage their personal image collections based on metadata and detection results.

USER STORIES:
1) As a User, I want to sign up so that I can access the application.
2) As a User, I want to verify my account using an OTP so that I can activate my account securely.
3) As a User, I want to log in to the platform so that I can use its features.
4) As a User, I want to remain logged in so I don't need to re-enter credentials repeatedly.
5) As a User, I want to reset my password so that I can recover access when I forget it.
6) As a User, I want to upload a photo with optional metadata so that it can be processed and stored.
7) As a User, I want the system to run face detection automatically so the image can be cropped properly.
8) As a User, I want age detection to run in the background so I get a result without waiting.
9) As a User, I want to retrieve the detected age for a specific photo.
10) As a User, I want to retrieve detected ages for multiple photos to avoid many separate API calls.
11) As a User, I want to delete one or more photos and their associated age records if desired.
12) As a User, I want to view all my uploaded photos and tags by email.
13) As a User, I want to retrieve photo content by ID so I can display it on the frontend.
14) As a User, I want to filter photos by email and tag to categorize and query them easily.
15) As a Developer, I want to containerize the age-detection-service so it can be deployed easily.

Microservices

name	description
Authentication Service	Handles user sign up, login, OTP verification, session management, and password reset to securely manage user access.
Photo Upload Service	Manages photo uploads along with optional metadata, storing images for further processing.
Face Detection Service	Performs automatic face detection on uploaded images to enable proper cropping and tagging.
Age Detection Service	Runs asynchronous age detection on photos in the background and provides age data retrieval APIs.
Photo Management Service	Allows users to view, filter, retrieve, and delete photos and their associated metadata and detection results.

Patterns

group_name	implementation_pattern	involved_microservices	explaination
Photo Processing	Database per Service	Photo Upload Service,Face Detection Service,Age Detection Service	Keeps photo data and metadata isolated for loose coupling and independent scalability
Bulk Data Retrieval	CQRS	Age Detection Service	Supports efficient bulk queries for detected ages
Efficient Querying	API Composition	Photo Management Service	Aggregates data from multiple services for efficient querying

7 point likert scale

1. Strongly disagree
2. Disagree
3. Somehow disagree
4. Neither agree nor disagree
5. Somehow agree
6. Agree
7. Strongly agree

18. Is the division in microservice correct with respect to the context and the user stories? *

1 2 3 4 5 6 7

☆ ☆ ☆ ☆ ☆ ☆ ☆

19. Is **API Composition Pattern** correctly applied? *

1 2 3 4 5 6 7

☆ ☆ ☆ ☆ ☆ ☆ ☆

20. Is **CQRS Pattern** correctly applied? *

1 2 3 4 5 6 7

☆ ☆ ☆ ☆ ☆ ☆ ☆

21. Is **Database per Service Pattern** correctly applied? *

1 2 3 4 5 6 7

☆ ☆ ☆ ☆ ☆ ☆ ☆

22. Observations

LuckyBets

SYSTEM DESCRIPTION:
LuckyBets will be a webapp where users can partecipate and play to various games. They can partecipate to decentralized lotteries and similars, which will be done using smart contracts deployed on the blockchain. In addition there will be a game like coin toss that will be done using P2P.

USER STORIES:
1. As a user, I want to connect my wallet, in order to access the games securely.
2. As a user, I want to see the number of players online, in order to gauge the activity level of the platform.
3. As a user, I want to select the amount to bet, in order to manage my risk while playing.
4. As a user, I want to choose between heads or tails, in order to personalize my betting experience.
5. As a user, I want to see the current bets placed by other players, in order to make informed decisions.
6. As a user, I want to see the signatures of the game round, in order to analyze that the outcome is fair.
7. As a user, I want to choose how many tickets to buy, in order to increase my chances of winning.
8. As a user, I want to confirm my transaction in the wallet popup, in order to ensure my purchase is secure.
9. As a user, I want to view the historical winnings, in order to see the potential rewards of the lottery.
10. As a user, I want to receive a confirmation of my ticket purchase, in order to have proof of my entry.
11. As a user, I want to see the next draw date, in order to plan my participation accordingly.
12. As a user, I want to buy a scratchcard, in order to participate in the game.
13. As a user, I want to use my mouse to scratch off the card, in order to reveal my prize.
14. As a user, I want to receive instant feedback on whether I won or lost, in order to enjoy the thrill of the game.
15. As a user, I want to have the option to buy multiple scratchcards at once, in order to increase my chances of winning.
16. As a user, I want to access a help section, in order to understand how to play each game.
17. As a user, I want to see current jackpots, so I know how much I can win.
18. As a user, I want to see the last wins, in order to know the latest won games.
19. As a user, I want to see platform stats, so that I can understand the platform’s activity before participating.

Microservices

name	description
Authentication Service	Handles user wallet connections and secure access to the platform.
Presence Service	Tracks and reports the number of players currently online.
Coin Toss Service	Manages the coin toss game including bet placement, choice selection, displaying current bets, and verifying game fairness.
Lottery Service	Handles decentralized lottery participation, ticket purchases, transaction confirmations, historical winnings, and draw scheduling.
Scratchcard Service	Manages scratchcard game operations including purchase, interactive scratching, instant result feedback, and bulk buying.
Statistics Service	Provides platform-wide statistics such as help content, current jackpots, recent wins, and overall platform activity.

Patterns

group_name	implementation_pattern	involved_microservices	explaination
Distributed Transactions	Saga	Lottery Service	The Saga pattern is ideal for managing complex, eventually consistent transactions like ticket purchases and blockchain confirmations in the Lottery Service.

7 point likert scale

1. Strongly disagree
2. Disagree
3. Somehow disagree
4. Neither agree nor disagree
5. Somehow agree
6. Agree
7. Strongly agree

23. Is the division in microservice correct with respect to the context and the user stories? *

1	2	3	4	5	6	7
☆	☆	☆	☆	☆	☆	☆

24. Is **Saga Pattern** correctly applied? *

1	2	3	4	5	6	7
☆	☆	☆	☆	☆	☆	☆

25. Observations

Utility of DALLE

Please answer the following questions related to the utility of the proposed approach.

Look at the demo of the tool to understand how it works.

26. Would you consider using DALLE to help design system architectures? *

Mark only one oval.

- Yes
- No
- Other: _____

27. Do you think DALLE could make your work easier or more efficient? *

Mark only one oval.

- Yes
- No
- Other: _____

28. Would you use DALLE to check or validate your design ideas? *

Mark only one oval.

- Yes
- No
- Other: _____

29. Do you find the suggestions provided by DALLE useful? *

Mark only one oval.

- Yes
- No
- Other: _____