Flight Path: AI-Powered Assessment and Personalized Recommendation

A System Plan Proposal Submitted to the Faculty
of the School of Computing and Information Technologies
Asia Pacific College, Makati City

In Partial Fulfillment of the Requirements of the subject
Introduction to Systems and Design for IT
SSYADD1

By

Justin Bryden G. Arroco
Don Victor L. Idos
John Michael O. Maala
Andrei Luis M. Torres

Adviser: Manuel L. Calimlim, Jr.
Consultant: Felino Calderon

I. Design Thinking

Stage 1 – Empathize:

Introduction

The first step of the design thinking process is to build empathy and develop a deep understanding of the people for whom we are designing the solution. For the “Flight Path” project, this meant engaging closely with the client, UpsWing!, to uncover their real needs, frustrations, and aspirations beyond what was written in initial project requirements. Our goal in this stage was not only to learn what they *say* they need, but also to understand what they *do, think, and feel* when navigating their current assessment process.

To accomplish this, our team conducted a series of interactions that served as in-depth client interviews. The first formal meeting was held through **Google Meet** with the UpsWing! CEO and representatives. During this session, we explored their current ESL assessment process, discussed its limitations, and aligned on the broader objectives of the project. The client emphasized recurring challenges such as inefficiency, reliance on manual evaluations, and scalability issues. They also stressed their commitment to academic rigor by referencing their use of CEFR standards, item analysis, and inter-rater reliability checks. This meeting set the foundation for understanding that the problem was not simply technical but also organizational and educational in nature.

Beyond the initial interview, our team also maintained **direct chat conversations** with the client. These discussions became a valuable channel for clarifying requirements, addressing changes, and confirming expectations in real time. This iterative exchange ensured that our understanding of the client’s pain points and goals remained aligned with their evolving needs.

Through these engagements, we not only documented the client’s processes but also began to empathize with the different personas involved: learners frustrated by slow placement, parents seeking clear reports before making payment decisions, tutors burdened by manual evaluations, and administrators struggling with operational bottlenecks. These insights highlighted the human side of the problem and provided the foundation for framing the root cause and defining what a successful solution should look like.

Client Interviews

To better understand the needs and challenges of the client, our team conducted a series of interviews and follow-up discussions with **UpsWing!’s CEO and representatives**. These interactions were carried out through both formal and informal channels, enabling us to gain a holistic perspective of the current ESL assessment process.

The **first formal interview** took place via **Google Meet**. This session allowed us to explore the existing placement and diagnostic system in detail. The client explained that their current process relies heavily on coordinators and admins, which makes it **time-consuming,**

inconsistent, and difficult to scale as enrollment grows. They emphasized that while efficiency is important, the solution must also uphold **academic rigor**, citing their reliance on CEFR standards, item analysis, and inter-rater reliability checks to maintain validity.

In addition to the formal meeting, we also engaged in **direct chat conversations** with the client through messaging platforms. These follow-up discussions were crucial for clarifying requirements, confirming project scope, and addressing updates or changes the client wanted to make as the project progressed. These exchanges gave us real-time insights into their expectations, such as the importance of distinguishing between adaptive placement tests and non-adaptive diagnostic tests, and their interest in integrating AI-driven speaking assessments.

From these interviews, several recurring themes emerged:

- **Inefficiency:** Current manual processes delay student evaluation and learning path assignment, frustrating students.
- **Scalability issues:** Reliance on admins and tutors creates bottlenecks as student numbers grow.
- **Consistency and rigor:** The system must ensure reliable, CEFR-aligned assessment results.
- **Stakeholder impact:** Tutors and admins face heavy workloads, while students risk mismatched placements.

FIGURE I: ONLINE GOOGLE MEET MEETING WITH UPSWING!

Identified Root Cause Analysis

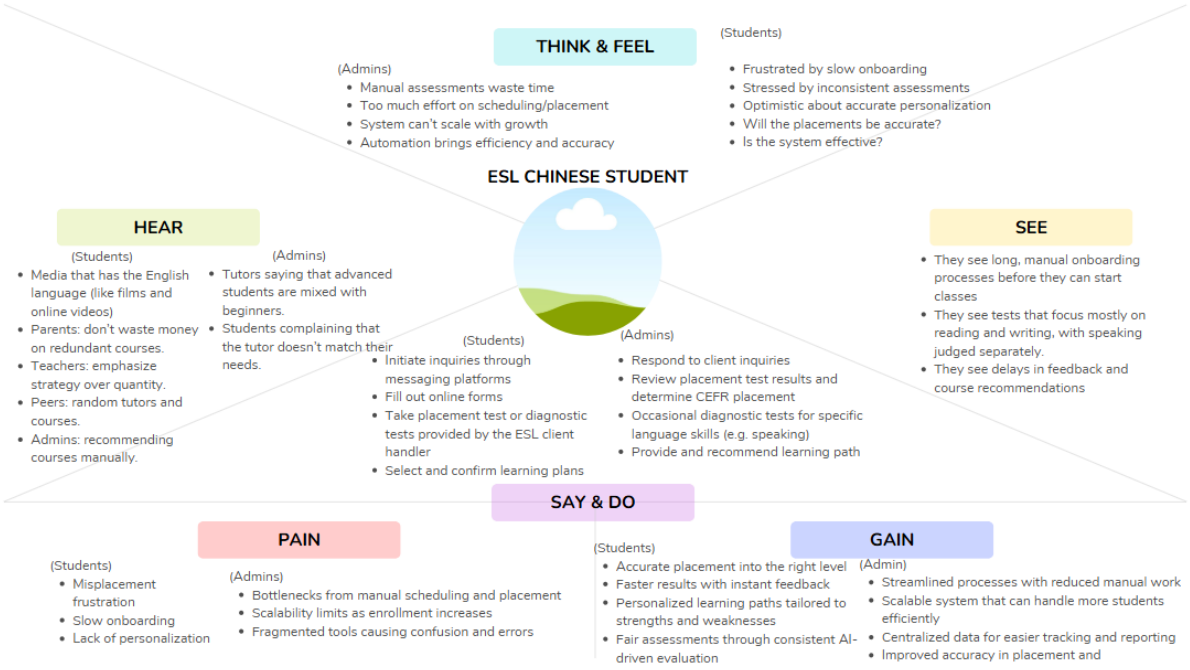
Analyzing the client's statements revealed the underlying cause driving this project.

Problem: UpsWing! needs to automate its assessment process.

- **Why #1?** The process is slow.
- **Why #2?** Because it is done manually.
- **Why #3?** Because admins must handle each assessment by hand, and they cannot keep up with the number of new requests.
- **Why #4? (Root Cause)** Because there is no existing automated system in place to support or replace the manual work.

Identified Customer Personas

FIGURE II: Empathy Map



The interviews and discussions also highlighted the following key user groups whose needs and pains must be addressed:

- **Students** – The end-user who takes the tests. Their experience is central to the project's goal of improving operational efficiency and accuracy. A fast, accurate assessment process is essential to their motivation and success.

- **Admins** – An internal user responsible for managing assessment workflows. Their pain points include repetitive manual tasks, bottlenecks, and coordination issues that slow down operations.
- **Tutors** – Another internal user who plays a role in reviewing students' performance and conducting trial classes. Their involvement in manual evaluation represents a significant labor cost and introduces subjectivity that the new system aims to minimize.

Key Open-Ended Questions

- **What is the core problem with the current system?**
  Answer: The current system is manual, labor-intensive, and inconsistent It is too slow to scale with growing demand and often fails to consistently assess students, leading to frustration and misaligned learning path.
- **What does a successful solution look like?**
  Answer: Success is an automated system that delivers computerized adaptive testing (CAT) for placement assessment and combined with AI-based diagnostic assessments for speaking and writing, in which ultimately outputs comprehensive recommendation feedback based on those assessments. This system should provide instant, accurate, CEFR-aligned results that guide learners into personalized learning paths.
- **What are the most important constraints and requirements?**
  Answer: The solution must function as a backend system that can integrate into the client's existing platform. It must clearly distinguish between adaptive placement tests and non-adaptive diagnostic tests, as this distinction is critical to the client's process.

Q&A Session with Client – UpsWing!

This session took place in a discord conversation.

Format: Q&A for Clarifications

- **Q1.** How is the accuracy of assessment tests usually evaluated in the company's processes? What frameworks (e.g., CEFR, IELTS standards) are you aligning with?
  A1. UpsWing evaluates accuracy through CEFR-aligned diagnostic rubrics across listening, speaking, reading, and writing (Pre-A1 to C2). Tests may also include IELTS-style tasks for comparability. Accuracy is ensured by:
- Conducting item analysis (difficulty index, distractor effectiveness)
- Performing inter-rater reliability checks for speaking/writing rubrics
- Running pilot assessments before rollout
- Aligning benchmarks with CEFR Companion Volume and academic references (Cambridge, ALTE, etc.)

- **Q2.** Is the system expected to have a “user feedback” functionality?
  A2. Yes. Feedback is integrated at multiple levels, including:
- Tutor feedback forms after every class
- Parent reaction buttons on lesson reports
- Monthly parent satisfaction check-ins (via WeChat/LINE/email)
- AI sentiment analysis on open-ended feedback
- Automatic flagging of low scores for review

- **Q3.** Will there be formal surveys, interviews, or usage analytics?
  A3. Yes. UpsWing applies triangulated feedback methods:
- Quarterly, pathway-based surveys for parents, learners, and tutors
- 1-on-1 interviews with parents/tutors (by HR or DSSI)
- Platform analytics (attendance rates, completion rates, reschedule frequency, dashboard engagement)
- Churn risk scoring from engagement/feedback data

- **Q4.** What is the current step-by-step ESL process at UpsWing!?
  A4.

1. **Client Inquiry Handling** – Contact via Messenger, WeChat, LINE, or website → Coordinator responds, explains program → Learner fills out online form → Briefing on diagnostic test.
2. **Placement Test & Analysis** – Learner completes LMS diagnostic test → CEFR level evaluated → Academic team drafts recommendation.
3. **Trial Class Scheduling** – Coordinator arranges 25-min trial → Materials sent via LMS → Tutor briefed.
4. **Trial Class Execution** – Tutor conducts session → Learner participates in speaking activities → Tutor submits evaluation.
5. **Level Recommendation & Enrollment** – Academic team finalizes CEFR placement → Parent selects plan/payment → Learner assigned tutor → Schedule confirmed.
6. **Class Delivery** – Regular ClassIn sessions → Tutor uses LMS materials → Homework + AI-reviewed tasks.
7. **Feedback & Reporting** – Tutor submits post-class feedback → Progress tracked in LMS → Monthly bilingual reports sent to parents.

- **Q5.** How will speaking and writing assessments work?
  A5.
- **Pronunciation Test:** Students read aloud sentences/paragraphs. Evaluates clarity, stress, intonation, vowel/consonant accuracy.
- **Speaking Test:** Topic prompts/role-play. Evaluates fluency, coherence, vocabulary, grammar, communicative effectiveness.
- **Writing Test:** Constructed responses (reports, reflections). Evaluated with AI scoring aligned to CEFR rubrics.

- **Q8.** Which tests will be adaptive vs. non-adaptive?
  A8.
- **Placement Assessment:** Adaptive (CAT) – used to determine CEFR level and general English skills.
- **Diagnostic Assessment:** Non-adaptive – focused on specific skill gaps after courses/lessons. Includes speaking/writing AI evaluations and non-adaptive multiple-choice for reading/listening.
- *Summary:* Placement = adaptive CEFR test. Diagnostics = non-adaptive gap analysis.

Insights Summary

The empathize stage confirmed that UpsWing!'s challenge goes beyond inefficiency; the deeper issue is that manual and inconsistent assessment processes lead to inaccurate placement, which in turn causes poor student engagement, dissatisfaction, and eventual churn. While efficiency and scalability are pressing business needs, the client emphasized that any solution must also preserve academic rigor and CEFR alignment to maintain credibility and learner trust.

Through client interviews, root cause analysis, and persona identification, we gained a clearer view of the needs of learners, parents, tutors, and administrators. Each group experiences the shortcomings of the current system differently, but all are connected by the need for a faster, more accurate, and scalable placement process.

These insights provide a strong foundation for the next stage of the design thinking process—Define—where we will synthesize the findings into a clear problem statement and identify opportunities for innovation.

I. Project Charter

Flight Path: AI-Powered Assessment and Personalized Recommendation

Purpose:

The Flight Path project was created to modernize and improve the English as a Second Language (ESL) assessment process at UpsWing!, an ESL education provider. Currently, their processes rely on time-consuming, manual evaluations that often result in inconsistent assessment decisions. Our goal is to solve these problems by introducing an AI-powered assessment system with AI-powered adaptive and diagnostic testing capability and personalized learning path recommendations. This will ensure that their onboarding and evaluation processes are faster, more accurate skill profiling, and tailored learning experiences that align with each learner's goals and proficiency level.

High-level project description

Flight Path is a modular backend system designed for integration with UpsWing!'s existing platform. It will include the ff:

- Assessment Module that are adaptive and non-adaptive. The adaptive tests will be based on psychometric models such as MIRT, for precise CEFR-aligned placement and non-adaptive tests will include more specific language skill evaluation such as speaking and writing assessments.
- Recommendation Engine that generates personalized learning paths based on the evaluation results.
- Administrative Tools for managing item banks, exporting reports, and monitoring assessment performance.

This system will be API-ready, allowing seamless integration with UpsWing!'s system while maintaining flexibility for future expansion.

High-level milestone schedule

TABLE I: HIGH-LEVEL MILESTONE SCHEDULE

Milestone	Estimated Week
Started Project Planning and Discovery, and Gathering of High-level Client Requirements	Apr 28 – May 11
User Research & Problem Definition Completed	May 12 – Jul 13
Finalization of Overall Project Requirements and Planning	Aug 4 – Aug 24
Finalization of System Architecture and API specs	Sep 8 – Sep 21
Development of Adaptive Test (includes prototype demonstration and iterative feedback gathering)	Sep 22 – Oct 26
Development of Non-Adaptive Diagnostic Tests for Specific Language Skills Evaluation (includes prototype demonstration and iterative feedback gathering)	Oct 27 – Nov 16
Implementation of Recommendation Engine (includes prototype demonstration and iterative feedback gathering)	Nov 17 – Dec 7
API Development of the Whole Backend module	Dec 7 – Jan 11
Internal Testing and Quality Assurance	Jan 12 - 18
Pilot Testing with Upswing Client	Jan 19 – Jan 25
Finalize Feedback Gathering and System Refinement	Jan 26 – Feb 8
Final Testing, Optimization, and Documentation	Feb 9 – Feb 14
Final Delivery and Project Closure	Feb 15 – Feb 25

Rough cost estimate

The total estimate is ₱102,200.00. This comprehensive budget is allocated to cover all anticipated expenses required to meet the project objectives, including labor, materials, and administrative overhead.

The budget is broken down into two primary categories:

Planned Labor Costs: ₱61,000.00. This covers an estimated 610 hours of work by the four-person project team, based on a standard rate of ₱100/hour.

Planned Material & Unit Costs: ₱41,200.00. This includes all non-labor expenses such as co-working space fees, software subscriptions, supplies, and a contingency reserve.

The budget is strategically allocated across the project lifecycle, with most resources dedicated to the Executing phase.

1.0 Initiation Phase – ₱23,000.00

2.0 Planning Phase – ₱24,000.00
3.0 Executing Phase – ₱41,200.00
4.0 Monitoring & Controlling Phase – ₱8,500.00
5.0 Closing Phase – ₱5,500.00

Stakeholders

- **Upswing! (Client / Organization)** – Provides budget, approves major decisions, sets priorities and requirements, gives feedback and recommendations.
- **Manuel L. Calimlim (Project Adviser)** – Provides academic supervision and general project guidance.
- **Felino Calderon (Project Consultant)** – Guides technical implementation, design choices, and overall project recommendations.
- **Tutors (Upswing ESL Tutors)** – End users who provide feedback on CEFR placement accuracy and diagnostic fairness.
- **Students (Upswing ESL Students)** – Primary end users; provide feedback on learning path recommendations and fairness of placement results.
- **W.A.S.D (Development Team, APC)** – Core builders of the FlightPath system; ensure functionality meets academic and client standards.
- **Admins (Upswing Academic/Operations Department)** – Use diagnostic and placement tools, review results, and provide ongoing feedback.
- **Instructor and School (Asia Pacific College, PBL Program)** – Oversee project progress, academic compliance, and documentation quality.
- **Testers** – Evaluate usability, provide performance feedback, and verify system stability before deployment.

Project manager: Andrei Luis M. Torres, Project Leader

Project manager's responsibilities:

- Oversee overall project coordination and lead team efforts.
- Ensure all documentation, written materials and deliverables are comprehensive and meet quality standards.
- Communicate with the project adviser on behalf of the team.
- Make sure the project is on track with the timeline.

Project manager's authority

- Lead project meetings and presentations to the client and adviser.
- Assign tasks to the dev team members.

- Guide the project workflow
- Manage which tasks to be prioritized
- Make final decisions with team input

Formal declaration of sponsor's support

Sponsor Name: Jose Eugenio L. Quesada

Position: Project-Based Learning Instructor

Support Statement:

As the instructor for the Project-Based Learning course, I support the FlightPath: AI-Powered Assessment and Personalized Recommendation project created by the student group. This project has the potential to modernize and improve the English as a Second Language (ESL) assessment and placement process for educational providers. It clearly demonstrates the students' ability to solve real-world problems through technology by addressing the inefficiencies of traditional evaluation methods. I encourage the group to complete this project with teamwork, effort, and creativity.

"Flight Path" Project Objectives

Primary Objective

Develop a modular, AI-assisted English language assessment and recommendation system to automate proficiency testing and generate personalized learning paths.

Specific Objectives

- Design and deploy an adaptive English assessment system within 6 months, achieving at least 90% accuracy in classifying learners according to CEFR levels.
- Integrate an AI-powered speaking assessment module that transcribes and scores learner responses, reducing manual evaluation time by at least 70%.
- Implement a data-driven recommendation engine within 4 months of system deployment, achieving at least 85% alignment with expert-curated recommendations.
- Deliver modular backend components, including the Adaptive Assessment Module, AI-Powered Speaking Assessment Module, Learner Recommendation Engine, and Administrative Utilities.

TABLE II: CATEGORIZED OBJECTIVE TABLE

Objective	Category	Measure
To design and deploy a modular, AI-assisted English language assessment and recommendation system for UpsWing!	Development	Delivery of modular backend components for an Adaptive Assessment Module, AI-Powered Speaking Assessment Module, Learner Recommendation Engine, and Administrative Utilities.
To design and deploy an adaptive English assessment system with at least 90% accuracy in classifying learners according to CEFR proficiency levels.	Accuracy	The system's classification accuracy will be verified against expert validation and must be at least 90%.
To integrate an AI-powered speaking assessment module that reduces manual evaluation time by at least 70%.	Efficiency	The reduction in time for manual evaluation will be measured after the module's integration and must be at least 70%.
To develop a data-driven recommendation engine that achieves at least 85% alignment with expert-curated recommendations.	Performance	Alignment with expert recommendations will be measured during pilot testing and must be at least 85%.
To develop a system that leverages psychometric models, transcribes and scores responses, and generates personalized learning paths.	Functionality	The system must include the core logic for a Computerized Adaptive Testing (CAT) system, AI-driven speaking assessment using APIs, and a recommendation engine.

Scope Statement

Project Goal and Objectives

The goal of the project is to develop a modular, AI-assisted English language assessment and recommendation system for UpsWing!. The system will automate proficiency testing and create personalized learning paths by accurately evaluating a learner's skills.

Project Boundaries

Within scope: The project includes the research, design, and development of reusable, modular backend components. The deliverables will be API-ready or plug-in compatible for UpsWing!'s existing platform. It includes:

- Adaptive Assessment Module
- AI-Powered Speaking Assessment Module
- Learner Recommendation Engine
- Standard Setting and CEFR Alignment Tools
- Administrative Support Utilities

Out of scope: The project does not include integration of the developed modules into the client's existing UI/UX, front-end, or data flows. This responsibility falls to the client-side developers. The project is limited to English assessments only and does not support multilingual handling. Front-end elements, UI components, and visualization of results are excluded.

Project Deliverables

- Adaptive Assessment Module
- AI-Powered Speaking Assessment Module
- Learner Recommendation Engine
- Standard Setting and CEFR Alignment Tools
- Administrative Support Utilities

Success Criteria

- The adaptive English assessment system will be designed and deployed within 6 months, using psychometric models to classify learners according to CEFR proficiency levels with at least 90% accuracy.
- The AI-powered speaking assessment module will be integrated by the end of the development phase, transcribing and scoring responses with at least a 70% reduction in manual evaluation time.
- The recommendation engine will be implemented within 4 months of the assessment system's deployment, producing personalized learning paths with at least 85% alignment to expert-curated recommendations during pilot testing.

Project Assumptions

- A demo item bank will be delivered, but full calibration using real learner data must be completed by the client.
- Accuracy and performance of the speech evaluation module depend on the reliability of external APIs.

Project Constraints

- Reliance on third-party APIs for speech evaluation.
- Exclusion of multilingual support.

- Exclusion of front-end/UI development.
- Integration into the client's existing system is the responsibility of the client-side developers.

Stakeholder Analysis

TABLE III: STAKEHOLDER ANALYSIS TABLE

Name	Department/Company	Position	Advisers	Objective, Requirements, Interests	Influence	Project Contribution	Resitance
Upswings	ESL Company	Organization/Client		Ensure project meets organizational goals and product quality.	High	Provides budget, approves major decisions, sets priorities and requirements, feedbacks and recommendations, and overall project information.	Concerns about scope and budget.
Manuel L. Calimlim Jr.	Asia Pacific College	Project Adviser		Ensure project meets organizational goals and product quality.	High	Provides supervision and academic guidance, and general project recommendations.	Concerns about overall project quality.
Felino Calderon	Asia Pacific College	Project Consultant		Ensures project meets technical requirements and scope.	High	Guides technical implementation and design choices, and general project recommendations.	Concerns about overall project quality.
Tutors	Upswing	ESL Tutors	Upswing Academic Department	Requires fair and accurate CEFR placement and clear diagnostic feedback that aligns with course preparation.	Medium	End users, and provides feedback.	Concerns about system accuracy and fairness.
Students	Upswing	ESL Students		Requires fair and accurate CEFR placement and diagnostic feedback, and receive appropriate learning path recommendations.	High	End users, and provides feedback.	Concerns about their overall student experience including fairness and appropriateness of recommended learning paths.
WASD	Asia Pacific College	Development Team	Project Adviser, Project Consultant	Deliver a functional FlightPath system that meets project requirements,	High	Core builders of the system (FlightPath).	Time/resource constraints, project technicality,

				satisfies client needs, and aligns with academic standards.			achievable project goals.
Admins	Upswing	Academic/Operation Department		Requires accurate placement and diagnostic results, and appropriate learning path recommendations.	High	End users, and provides feedbacks.	Inaccurate assessment system and recommendation engine.

Tech Stack

Backend & Framework

- Programming Language: Python 3.11
- Web Framework: FastAPI
- ASGI Server: Uvicorn

Database & Data Management

- Database: MySQL 8
- ORM: SQLAlchemy (async)
- Schema Migrations: Alembic
- Configuration: dotenv (for environment variables)

AI, Data Science, & Core Libraries

- AI Engine (Core): MirtCAT (Multidimensional Item Response Theory Computerized Adaptive Testing) engine
- AI Service API: Gemini API (for speech-to-text, writing evaluation, and NLP)
- Numerical Libraries:
    - NumPy
    - SciPy
- Data Analysis: Pandas

DevOps & Tools

- Containerization: Docker and Docker Compose
- Version Control: GitHub
- API Testing & Documentation:
    - Postman
    - Swagger Docs / Swagger UI

IV. Diagrams

FIGURE III: USE CASE DIAGRAM

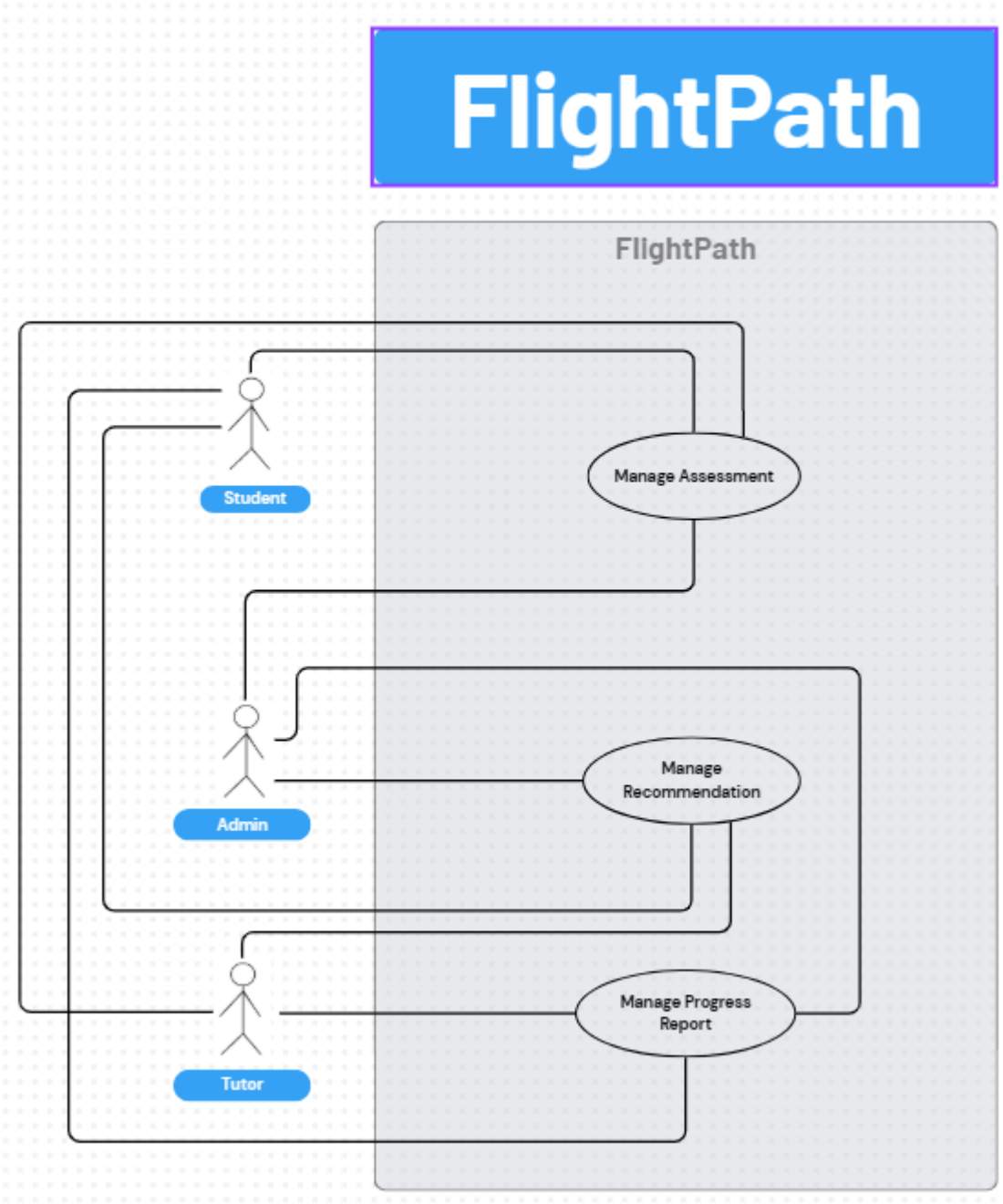


TABLE IV: TEST CASE TABLE

Test Case ID	Use Case ID	Test Case Name	Test Case Description
TC-01	UC-01	Admin Creates Valid Assessment (Basic Flow)	Verify the Admin can successfully create a new assessment by providing all valid parameters and item IDs.
TC-02	UC-01	Admin Creates New Item During Assessment Creation (Alt Flow)	Verify the Admin can successfully create a new item via the alternative flow, save it to the bank, and use its ID in the new assessment.
TC-03	UC-01	Admin Cancels Assessment Creation (Alt Flow)	Verify that if the Admin invokes the "Cancel" operation, the assessment data is not saved.
TC-04	UC-01	Admin Creates Assessment with Missing Data (Exception)	Verify the system outputs a validation error and does not save if the Admin tries to create an assessment with missing required parameters.
TC-05	UC-02	Admin Assigns Assessment to Single Student (Basic Flow)	Verify the Admin can provide an assessment ID and a single student ID to successfully confirm the assignment.
TC-06	UC-02	Admin Assigns Assessment to Group (Basic Flow)	Verify the Admin can provide an assessment ID and a student group ID to successfully confirm the assignment to all students in that group.
TC-07	UC-02	Admin Assigns When No Assessments Exist (Exception)	Verify the system outputs an appropriate error message if the Admin tries to assign an assessment but no active assessments are available.
TC-08	UC-02	Admin Assigns to Non-Existent Student (Exception)	Verify the system outputs an error if the Admin attempts to assign an assessment to a student ID that does not exist.
TC-09	UC-03	Student Completes Placement Assessment (Basic Flow)	Verify a Student can request and successfully complete a "Placement" assessment, and the results are stored.
TC-10	UC-03	Student Completes Writing Assessment (Basic Flow)	Verify a Student can request, complete, and submit the data for a "Writing" assessment, and the results are stored.
TC-11	UC-03	Student Completes Speaking Assessment (Basic Flow)	Verify a Student can request, provide answer data for, and submit a "Speaking" assessment, and the results are stored.
TC-12	UC-03	Admin Tests an Assessment (Basic Flow)	Verify an Admin can request and execute any assessment type to test/verify its functionality.
TC-13	UC-03	Student Exits Assessment Mid-Way (Alt Flow)	Verify that if a Student's session terminates mid-assessment, the system automatically saves their state.
TC-14	UC-03	Student Resumes Assessment (Alt Flow)	Verify a Student can re-initiate an assessment they exited mid-way and resume or restart it.

TC-15	UC-03	Student Re-records Speaking Answer (Alt Flow)	Verify a Student can invoke the re-record command during a "Speaking" assessment to replace a previous audio response before submission.
TC-16	UC-03	No Microphone Detected for Speaking Test (Exception)	Verify the system detects the absence of a working microphone (e.g., failed hardware check) and outputs an error when a Student starts a "Speaking" assessment.
TC-17	UC-03	Internet Failure During Submission (Exception)	Verify the system outputs an error message if the connection fails during the submission process.
TC-18	UC-03	Assessment Items Fail to Load (Exception)	Verify the system outputs an error message if the assessment items cannot be retrieved from the database.
TC-19	UC-03	Speaking/Writing Submission Fails (Exception)	Verify the system outputs an error message if the submission fails due to a file processing error or timeout.
TC-20	UC-04	Admin Edits Assessment Details (Basic Flow)	Verify an Admin can provide an existing assessment ID and modified parameters (e.g., title), and the changes are saved.
TC-21	UC-04	Admin Cancels Edit Assessment (Alt Flow)	Verify that if an Admin invokes the "Cancel" operation while editing, no changes are saved to the assessment.
TC-22	UC-04	Admin Edits Assessment with Invalid Data (Exception)	Verify the system outputs a validation error if the Admin tries to save invalid data in a modified parameter.
TC-23	UC-04	Admin Edits Assessment in Active Use (Exception)	Verify the system prevents an Admin from saving changes to an assessment that is actively in-progress by a student.
TC-24	UC-04	Admin Edits Non-Existent Assessment (Exception)	Verify the system outputs an error if the Admin attempts to edit an assessment ID that does not exist.
TC-25	UC-05	Admin Archives an Assessment (Basic Flow)	Verify an Admin can provide an assessment ID, invoke "Archive," and confirm the action, resulting in the assessment status being "Archived".
TC-26	UC-05	Admin Archives Assessment in Use (Exception)	Verify the system prevents an Admin from archiving an assessment that is currently assigned or in-use by students.
TC-27	UC-06	Student Reviews Own Assessment History (Basic Flow)	Verify a Student can request "Assessment History" and receive a list of their own completed assessments.
TC-28	UC-06	Student Filters Assessment History (Basic Flow)	Verify a Student can use filter and sort parameters in their history request, and the returned results are correct.
TC-29	UC-06	Admin Reviews Student's History (Alt Flow)	Verify an Admin/Tutor can request the assessment history for a specific student ID and receive that student's data.

TC-30	UC-06	User Reviews History for New Student (Exception)	Verify the system outputs a "No assessment history is found" message when requesting the history of a student who has none.
TC-31	UC-07	Student Exports Own Assessment History (Basic Flow)	Verify a Student can use filter parameters and invoke "Export" to generate and receive their own assessment history file.
TC-32	UC-07	Admin Exports Student's History (Alt Flow)	Verify an Admin/Tutor can request an export of a specific student's history and receive the correct file.
TC-33	UC-07	User Exports Empty History (Exception)	Verify the system outputs a "No history found" message if the user tries to export an empty history report.
TC-34	UC-07	Export File Generation Fails (Exception)	Verify the system outputs an error message if the file generation process fails on the server.
TC-35	UC-07	Export File Download Fails (Exception)	Verify the system outputs an error message if the file transmission fails.
TC-36	UC-08	Automatic Recommendation Generation (Basic Flow)	Verify that after a Student completes an assessment, the system automatically sends results and stores generated recommendations.
TC-37	UC-08	Admin Manually Triggers Recommendation (Alt Flow)	Verify an Admin can provide a student's assessment ID and manually trigger the "Generate Recommendations" process.
TC-38	UC-08	Recommendation Engine Fails (Exception)	Verify the system logs an error if the recommendation engine fails or times out during generation.
TC-39	UC-08	Recommendation from Corrupted Results (Exception)	Verify the system handles an error if the assessment results are missing or corrupted and cannot be analyzed.
TC-40	UC-09	Student Reviews Own Recommendations (Basic Flow)	Verify a Student can request the recommendations for a completed assessment and receive the associated data.
TC-41	UC-09	Admin Reviews Student's Recommendations (Alt Flow)	Verify an Admin/Tutor can request the recommendations for a specific student's assessment and receive the data.
TC-42	UC-09	User Reviews Non-Existent Recommendations (Exception)	Verify the system outputs a "No recommendations are found" message if the user requests recommendations that haven't been generated.
TC-43	UC-10	Admin Overrides Recommendation (Basic Flow)	Verify an Admin can provide a student's recommendation ID, invoke "Override," submit new items, and save the replacement.
TC-44	UC-10	Admin Overrides with Invalid Items (Exception)	Verify the system outputs an error if the Admin tries to save an override with item IDs that are invalid or non-existent.

TC-45	UC-10	Admin Saves Empty Override (Exception)	Verify the system outputs an error and does not save if the Admin attempts to save an empty recommendation list as an override.
TC-46	UC-11	Student Reviews Own Progress Report (Basic Flow)	Verify a Student can request "Progress Report" and receive their own analytics and trends.
TC-47	UC-11	Student Filters Progress Report (Basic Flow)	Verify a Student can use filter parameters in their progress report request, and the returned data is correct.
TC-48	UC-11	Admin Reviews Student's Progress Report (Alt Flow)	Verify an Admin/Tutor can request the real-time progress report for a specific student ID.
TC-49	UC-11	Analytics Generation Fails (Exception)	Verify the system outputs an error message if the analytics engine fails or times out.
TC-50	UC-11	User Reviews Report for New Student (Exception)	Verify the system outputs a "No data is found" message when trying to retrieve a report for a student with no assessment results.
TC-51	UC-12	Student Exports Own Progress Report (Basic Flow)	Verify a Student can request an export of their progress report, using filters, and successfully receive the file.
TC-52	UC-12	Admin Exports Student's Progress Report (Alt Flow)	Verify an Admin/Tutor can request an export of a specific student's progress report and receive the correct file.
TC-53	UC-12	Analytics Fail During Export (Exception)	Verify the system outputs an error if the analytics generation fails during the export process.
TC-54	UC-12	User Exports Report for New Student (Exception)	Verify the system outputs a "No data is found" error if attempting to export a report for a student with no data.
TC-55	UC-12	Export File Generation Fails (Exception)	Verify the system outputs an error message if the report file generation fails on the server.
TC-56	UC-12	Export File Download Fails (Exception)	Verify the system outputs an error message if the report file transmission fails.

FIGURE IV: DFD LEVEL 0

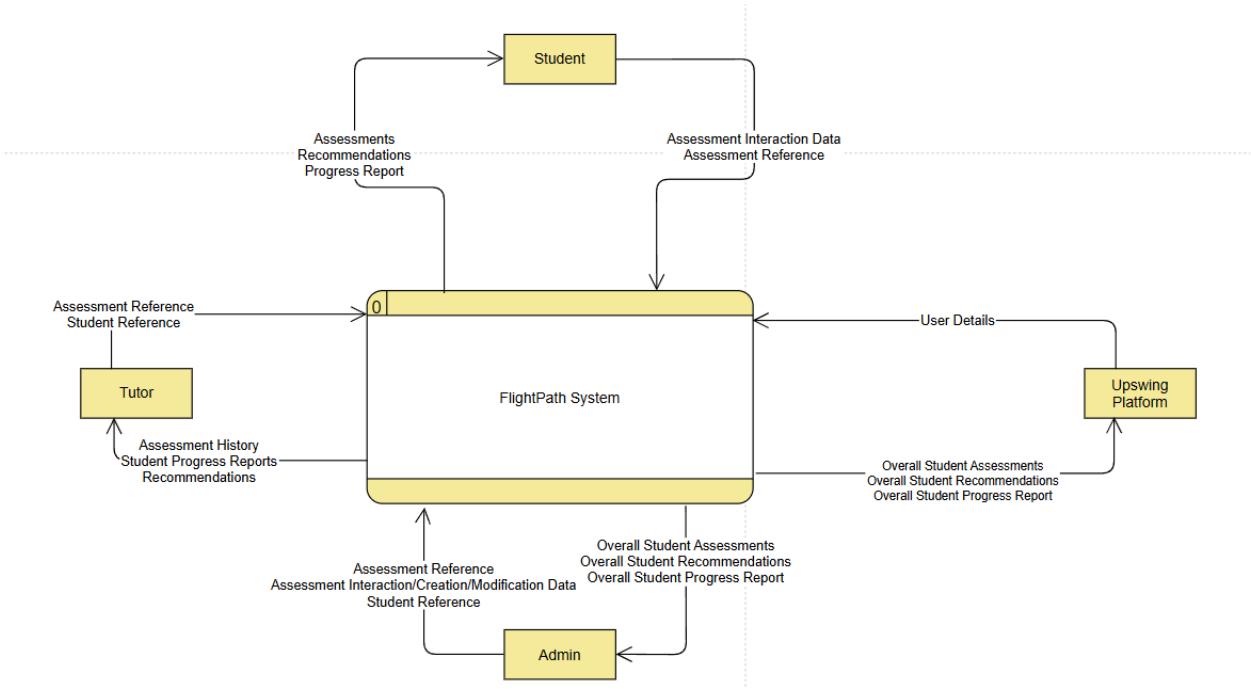


FIGURE V: DFD LEVEL 1

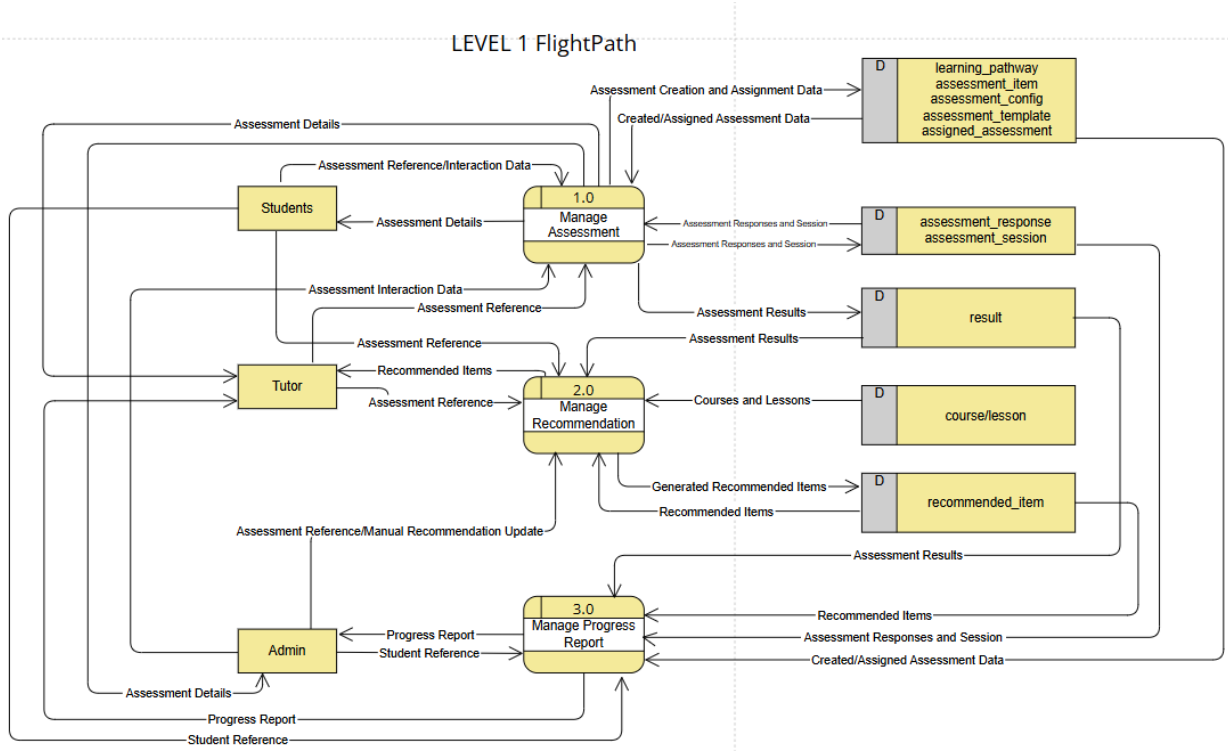


FIGURE VI: DFD LEVEL 2 – 1.0

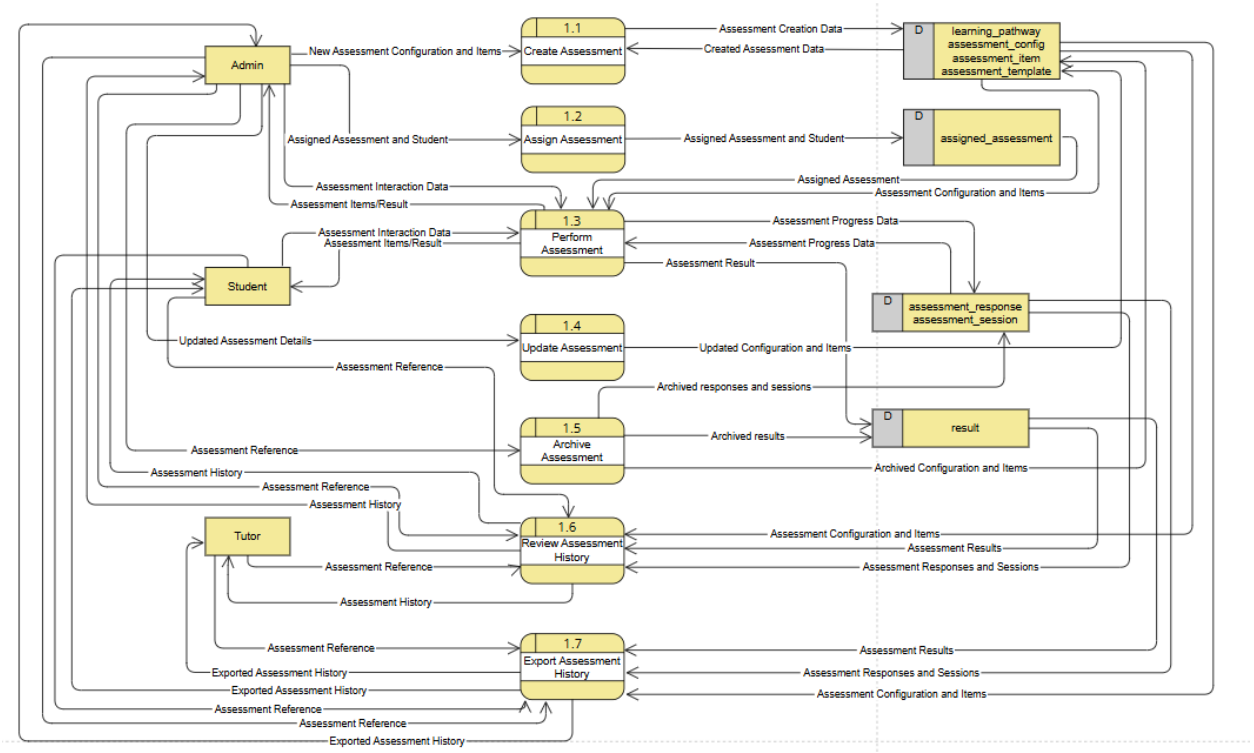


FIGURE VII: DFD LEVEL 2 – 2.0

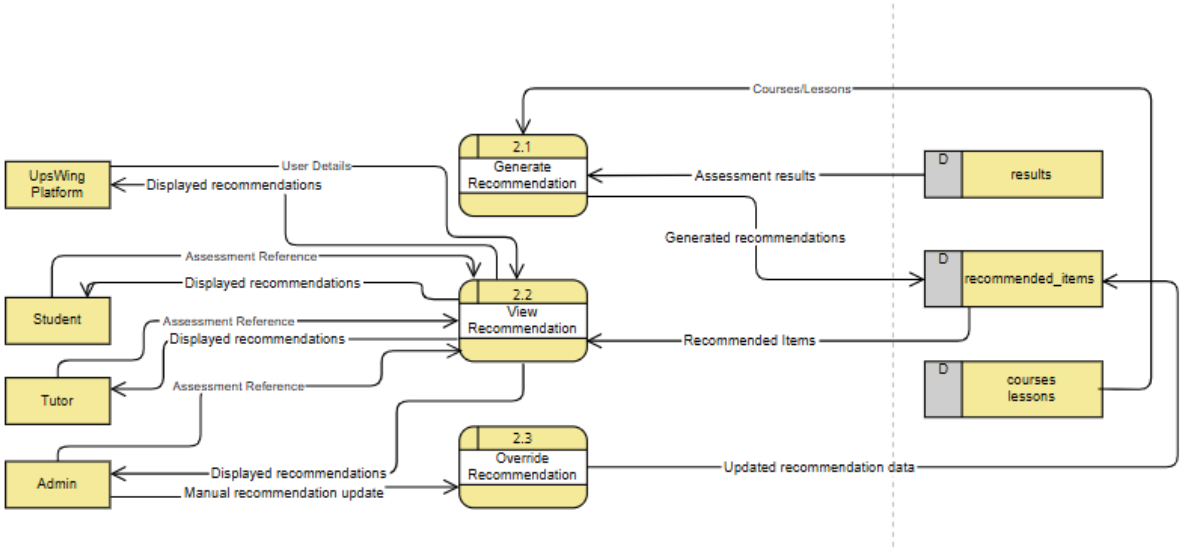


FIGURE VIII: DFD LEVEL 2 – 3.0

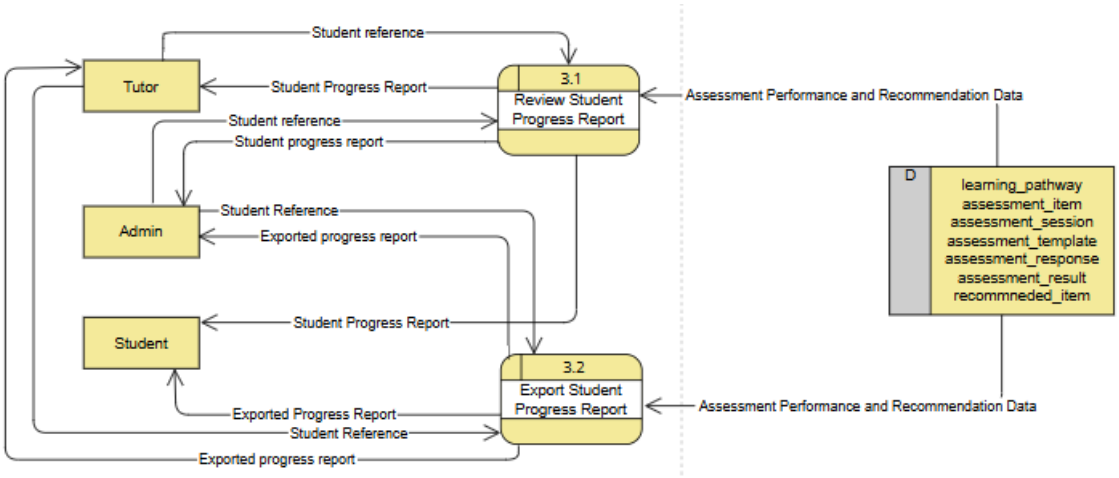


FIGURE IX: DFD LEVEL 3 – 1.3.x

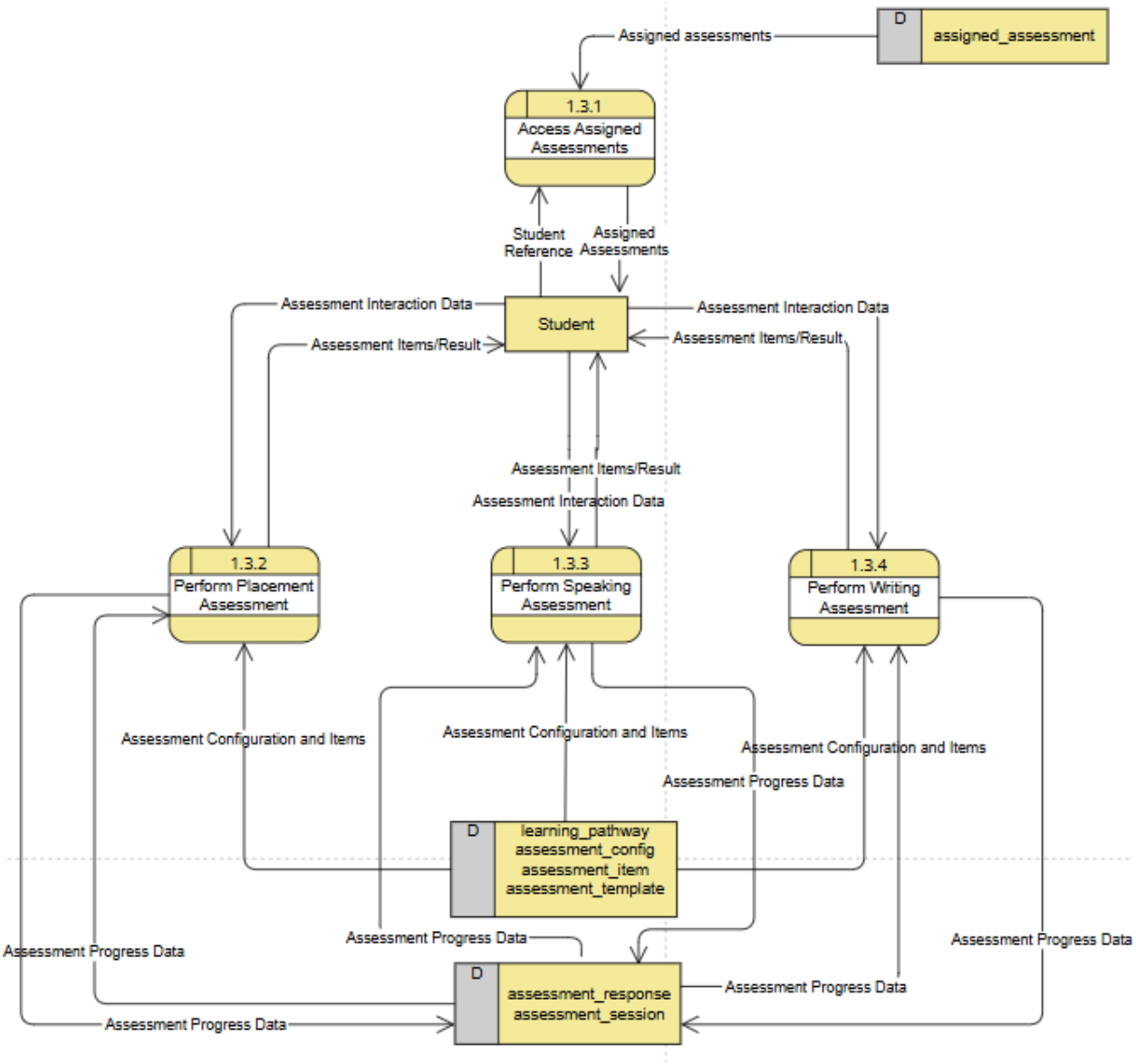


FIGURE X: DFD LEVEL 3 – 1.3.x

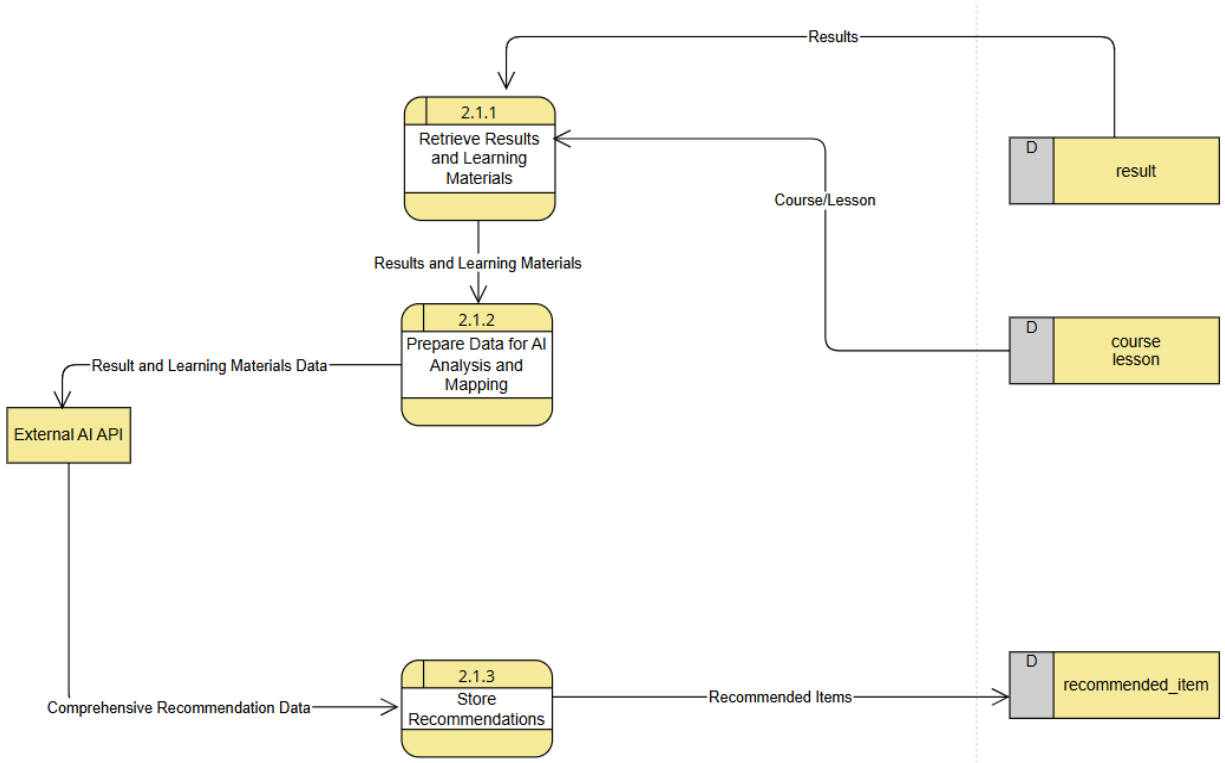


FIGURE XI: DFD LEVEL 4 – 1.3.2.x

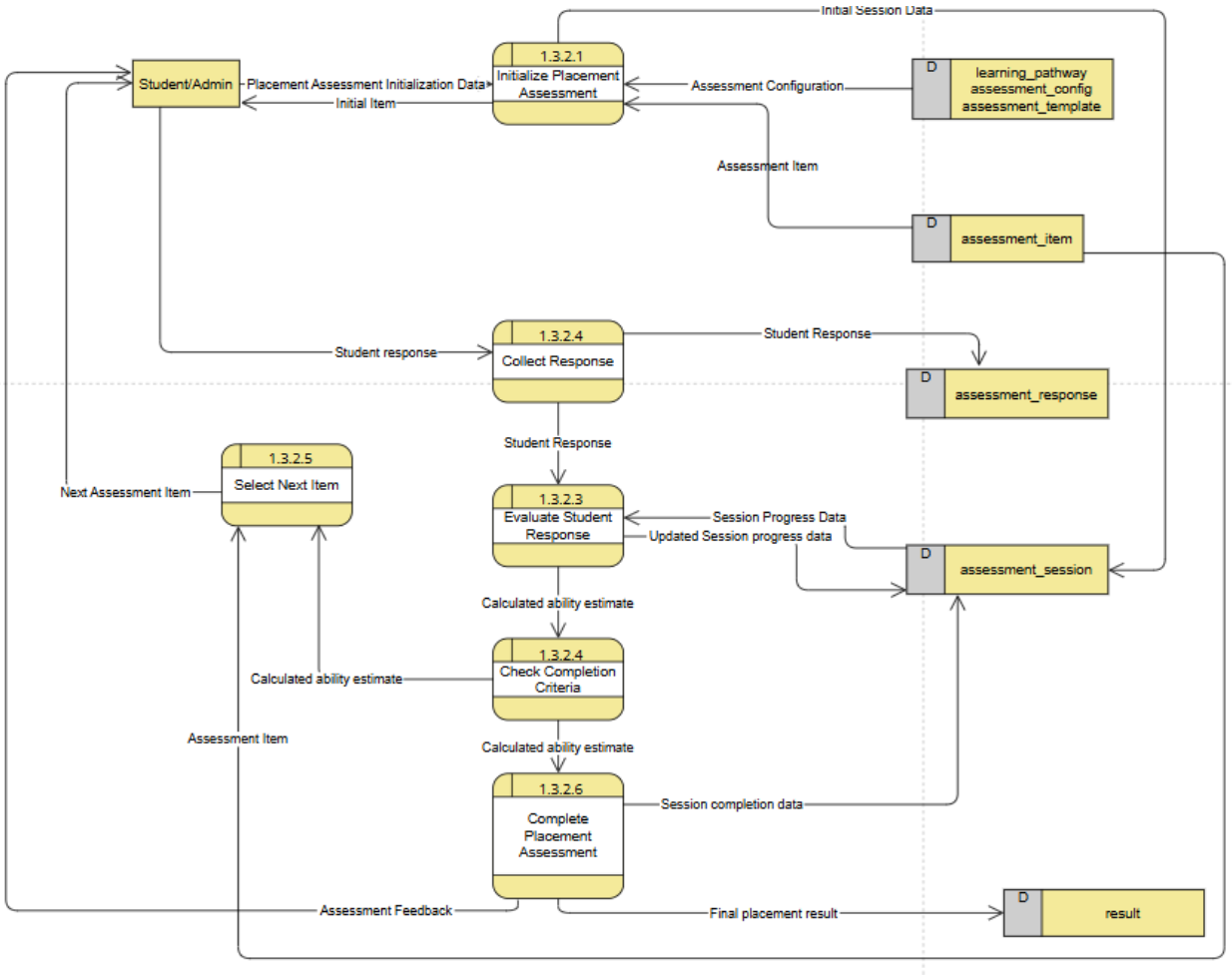


FIGURE XII: DFD LEVEL 4 – 1.3.3.x

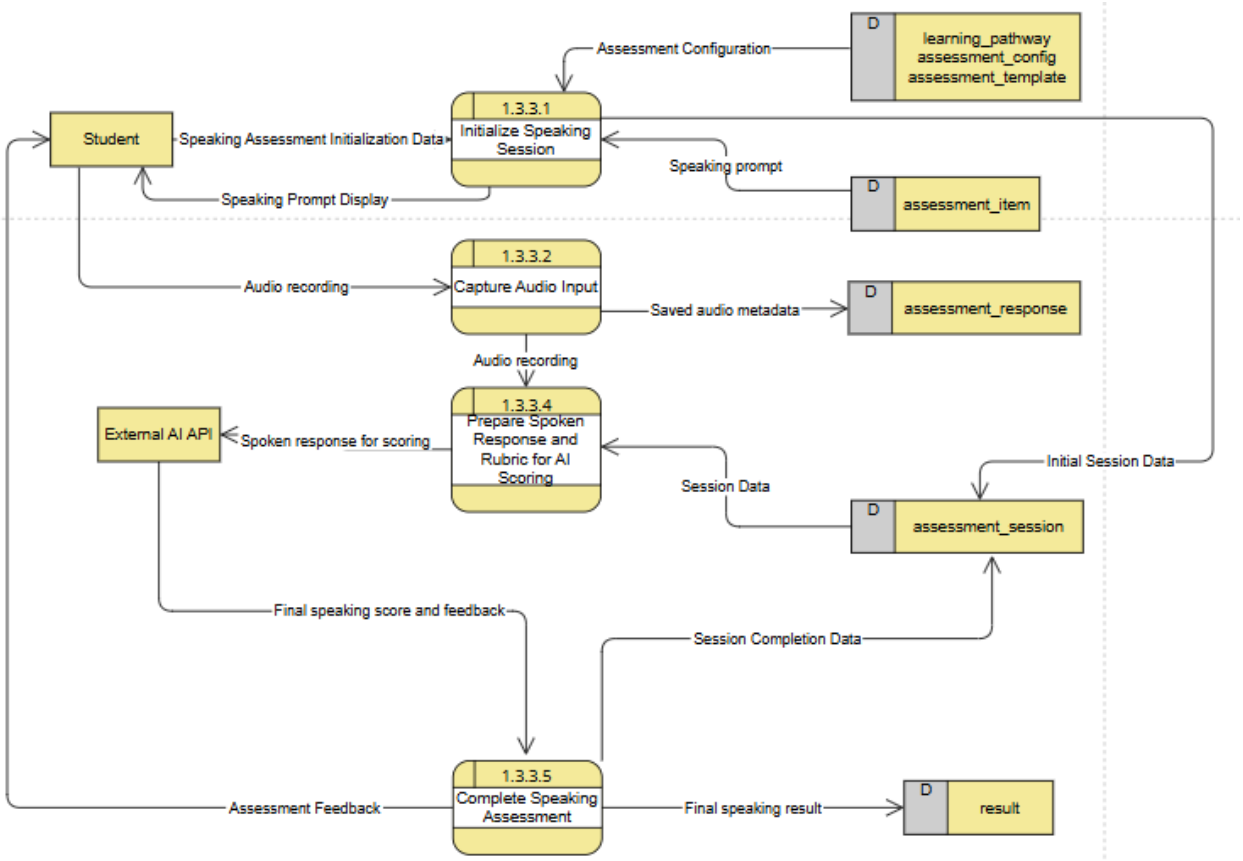


FIGURE XIII: DFD LEVEL 4 – 1.3.4.x

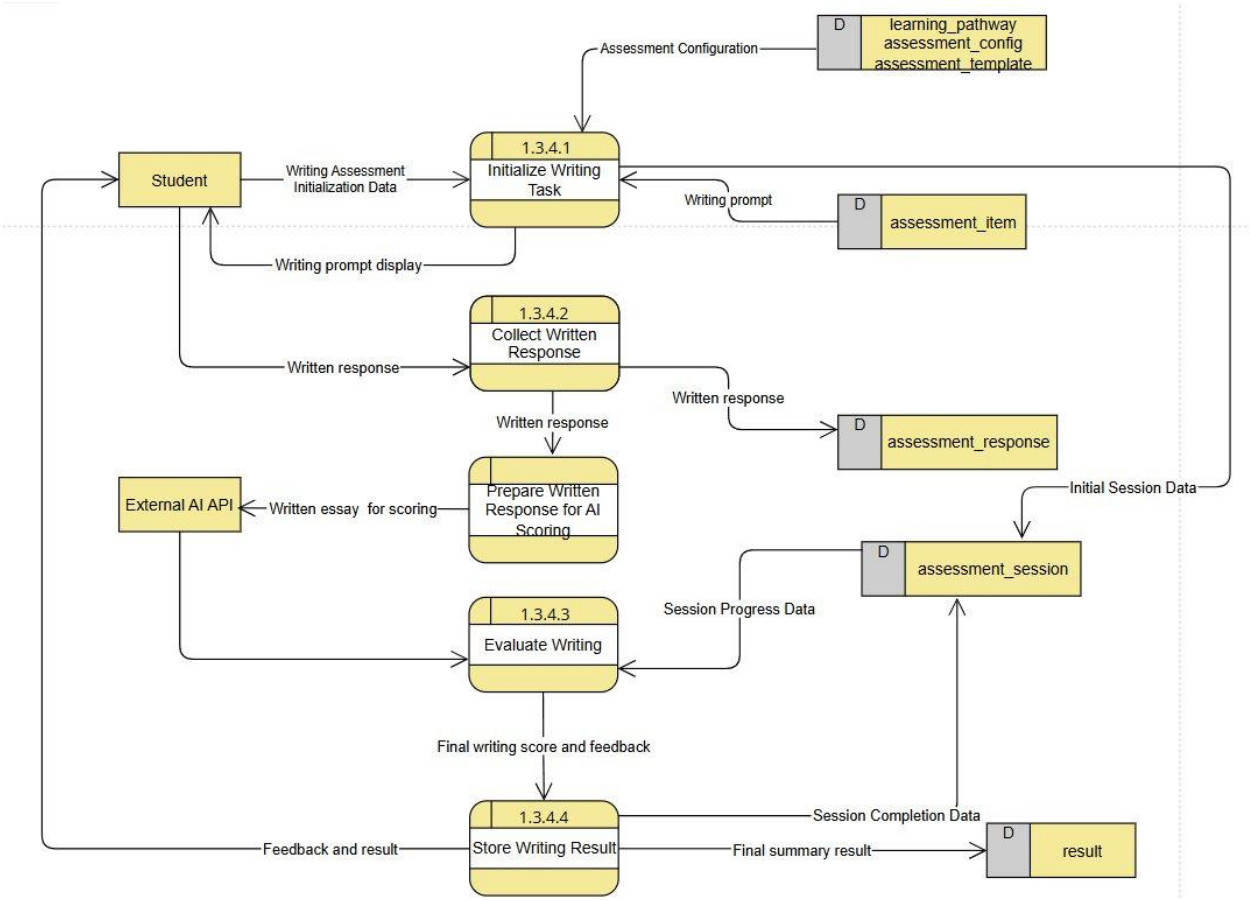
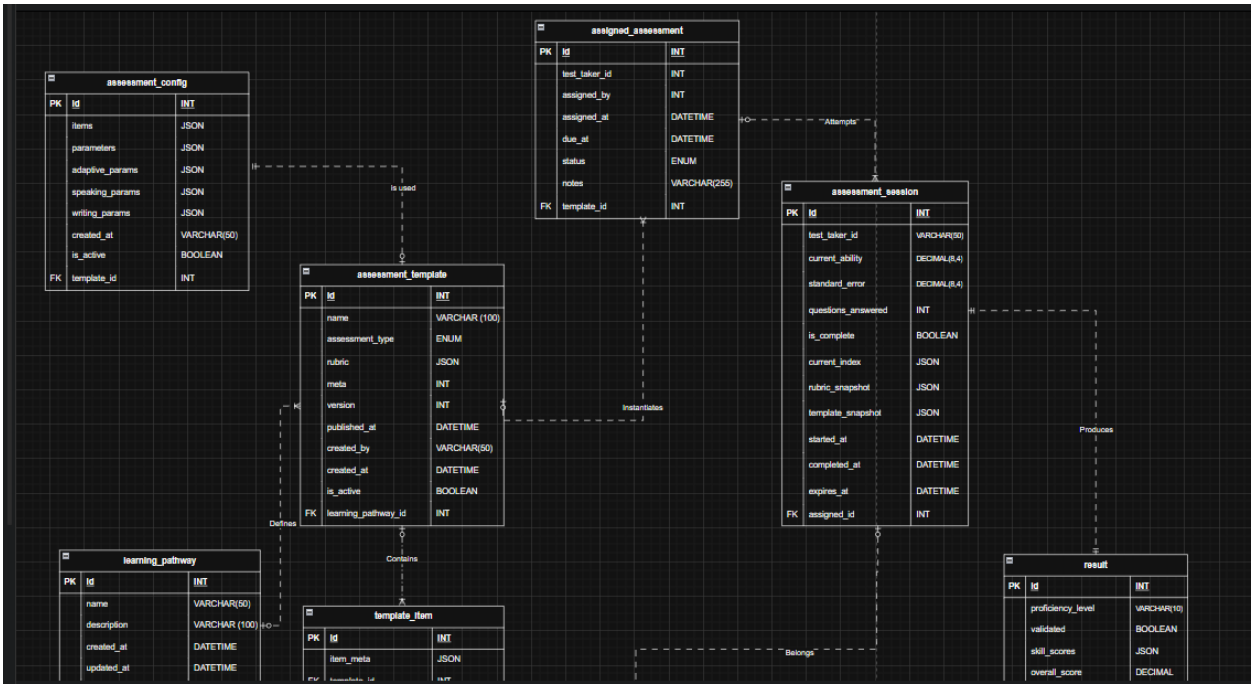
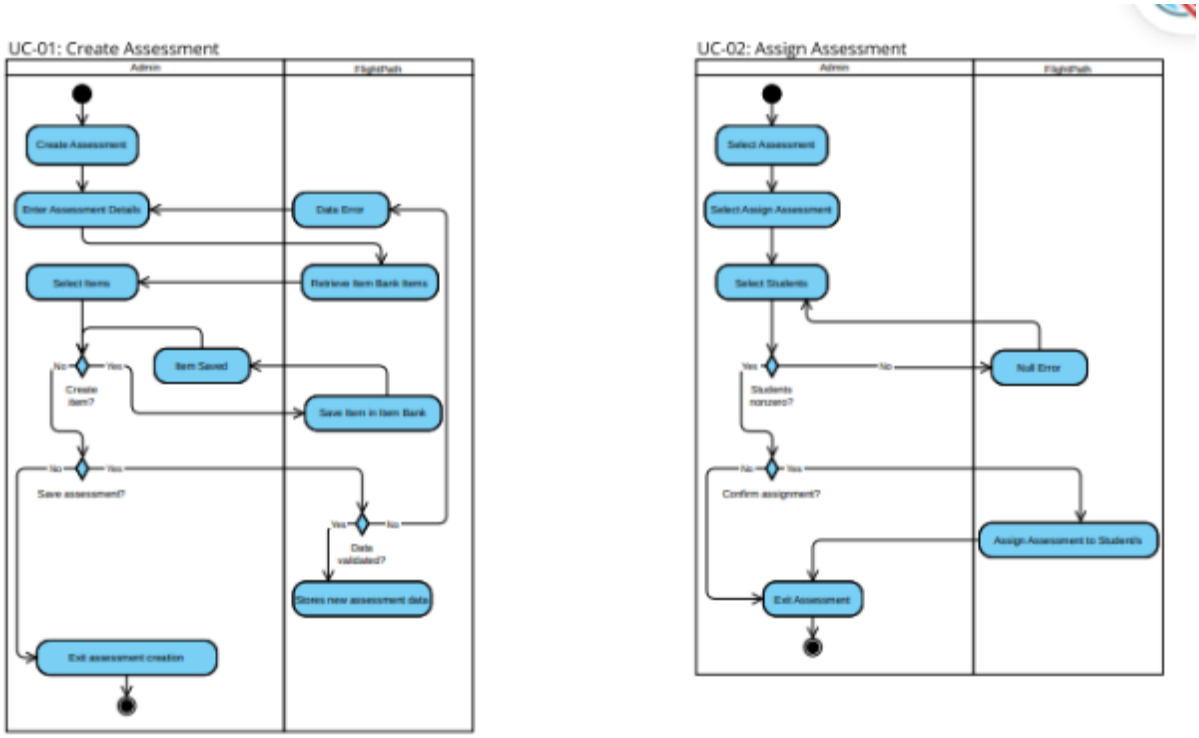


FIGURE XIV: ERD DIAGRAM



[Link to ERD Diagram]

FIGURE XV: ACTIVITY DIAGRAM



Activity Diagram Link

4.1 Use Case Name: Create Assessment

Use Case ID	**UC-01**
Actors	**Admin**
Description	Enables **admins** to create new assessments.
Preconditions	**Admin** must be logged in and have access to the system with sufficient permissions.
Basic Flow	1. Admin selects “Create Assessment”. 2. Admin enters details such as title, type, skill coverage, CEFR alignment, learning pathway, and other parameters. 3. System retrieves items from the database. 4. Admin selects items. 5. Admin confirms and saves the assessment. 6. System validates and stores assessment data.

Postconditions	• A new assessment is added to the database and ready for assignment to students. • The action is logged.
Alternative Flows	**Step 4a** 4a. Admin creates new item with its own parameters. 5a. Item saves in the item bank, which can now be selected. 6a. Go back to Step 4 of Basic Flow. **Anytime** • Admin cancels the creation process.
Assumptions	• The system is online and accessible. • The item bank is populated.
Exceptions	• Validation fails (e.g. missing or invalid parameters)

4.2 Use Case Name: Assign Assessment

Use Case ID	**UC-02**
Actors	**Admin**
Description	Allow an **admin** to assign a specific assessment to a student or a group of students.
Preconditions	The **admin** must be logged in and have sufficient permissions.
Basic Flow	1. The admin selects an assessment. 2. The admin selects the "Assign Assessment" option. 3. The admin selects a student or group of students. 4. The admin confirms the assignment. 5. The system assigns the assessment to the student.
Postconditions	• The assessment is assigned to the student(s). • The action is logged.
Assumptions	• There are active assessments available in the system. • There are registered students in the system.
Exceptions	• No active assessments are available. • The selected student does not exist.

4.3 Use Case Name: Perform Assessment

Use Case ID	**UC-03**
Actors	**Student, Admin**

Description	Allows the **student** to take an English assessment that measures proficiency and CEFR level with three different types of assessment: **Placement**, **Speaking**, and **Writing**. Allows the **admin** to test/verify the assessment.
Preconditions	The **user** is logged in and has access to the UpsWing! Platform
Basic Flow	1. User selects "Perform Assessment" option. 2. System retrieves assigned assessments. 3. Student chooses which assessment to perform. (Note: All three paths are mutually exclusive) **If the assessment type is "Placement":** 3a. System initializes placement assessment. 4a. System sends an assessment item. 5a. User answers the assessment item. 6a. System processes the answer. 7a. System checks if completion criteria are met ("max items reached" or "ability estimate stable"). System automatically completes the assessment. 8a. Else, system selects the next item and jumps back to 4a. **If the assessment type is "Writing":** 3b. 4b. System initializes writing assessment. 5b. System sends all assessment items. 6b. User types written answers. 7b. User submits the answers. 8b. System processes the answers. **If the assessment type is "Speaking":** 4c. System initializes speaking assessment. 5c. System sends all assessment items. 6c. User speaks each answer into the microphone. 7c. User submits the answers. 8c. System processes the answers.

	All branches merge: 8. 9. System stores results and sends a completion message.
Postconditions	• Student's assessment responses and assessment results are saved • Results are ready for recommendation generation. • The action is logged.
Alternative Flows	**If the user exits mid-assessment:** 1. Student exits mid-assessment. 2. System saves the state automatically. 3. Student can resume or restart the assessment. (return to Step 3a/3b/3c of Basic Flow) **From Step 5c -> Step 5d:** • User re-records the audio response. (return to Step 5c of Basic Flow)
Assumptions	• The user has a basic understanding of how to answer items online. • The user has a working microphone for the "Speaking" type of assessment. • The system is online and accessible. • The user has a stable internet connection, especially for submitting "Speaking" answers.
Exceptions	• System fails to detect a working microphone for the "Speaking" assessment. • User's internet connection fails during submission of the assessment. • Assessment items fail to load. • The "Speaking" or "Writing" submission fails (e.g., file upload error, timeout).

4.4Use Case Name: Update Assessment

Use Case ID	**UC-04**
Actors	**Admin**
Description	Allows the **admin** to modify assessment details or replace test items.
Preconditions	**Admin** must be logged in and have access to the system with sufficient permissions.
Basic Flow	1. Admin selects an existing assessment. 2. System retrieves assessment data. 3. Admin clicks the Edit button.

	4. Admin modifies fields (title, difficulty, item set, other parameters). 5. Admin saves the changes. 6. System validates and updates assessment data.
Postconditions	• Assessment record updated. • The action is logged.
Alternative Flows	**Step 5a.** • Admin selects "Cancel" and discard any changes.
Assumptions	• The system is online and accessible. • The assessment being updated is not currently in-progress by any students.
Exceptions	• Validation fails on modified fields. • Admin attempts to edit an assessment that is actively in-use. • The selected assessment does not exist.

4.5Use Case Name: Archive Assessment

Use Case ID	**UC-05**
Actors	**Admin**
Description	Enables **admins** to remove outdated assessments from active use without deletion.
Preconditions	**Admin** must be logged in and have access to the system with sufficient permissions.
Basic Flow	1. Admin selects an existing assessment. 2. Admin clicks the Archive button. 3. System prompts confirmation. 4. Admin confirms action. 5. System validates assessment. 6. System archives the updates the status as Archived.
Postconditions	• Assessment is marked as Archived and hidden from active lists. • The action is logged.
Assumptions	• The assessment is outdated, not in use. • The system is online and accessible.
Exceptions	• Admin attempts to archive an assessment that is assigned or currently in-use.

4.6Use Case Name: Review Assessment History

Use Case ID	**UC-06**

Actors	**All users**
Description	Allows **students**, **admins**, and **tutors** to review a record of assessments of a specific **student**.
Preconditions	The **user** is logged in and has access to the UpsWing! platform
Basic Flow	**Student** 1. Student selects “Assessment History”. 2. System retrieves records. 3. Student applies filters and sorts. 4. Student reviews History and clicks on a specific Assessment. 5. System sends details about the Assessment.
Postconditions	• Assessment history file was reviewed. • The action is logged.
Alternative Flow	**If the user is Admin or Tutor:** 1a. The user selects a Student. 2a. Go to Basic Flow Step 1.
Assumptions	• The **student** has existing history to review.
Exceptions	• No assessment history is found for the student.

4.7Use Case Name: Export Assessment History

Use Case ID	**UC-07**
Actors	**All users**
Description	Allows **students**, **admins**, and **tutors** to export a record of assessments of a specific **student**.
Preconditions	The **user** is logged in and has access to the UpsWing! platform
Basic Flow	**Student** 1. Student selects “Assessment History”. 2. System retrieves records. 3. Student applies filters. 4. Student clicks Export. 5. System generates file and downloads to local storage.
Postconditions	• Assessment history file is downloaded (CSV or PDF). • The action is logged.
Alternative Flow	**If the user is Admin or Tutor:** 1a. The user selects a Student. 2a. Go to Basic Flow Step 1.
Assumptions	• The **student** has existing history to download.

	• The user's device has software capable of reading the file.
Exceptions	• No history found. • File generation fails. • File download fails.

4.8Use Case Name: Generate Recommendation

Use Case ID	**UC-08**
Actors	**Student, Admin**
Description	Generates tailored recommendations for **students** based on the results of their assessment. Allows **admin** to manually trigger the process.
Preconditions	• The **user** is logged in and has access to the UpsWing! platform • The student completes an assessment.
Basic Flow	**Student** 1. System automatically sends results to the recommendation engine. 2. System analyzes results and generates recommendations. 3. System stores recommendations for reviewing.
Postconditions	• Recommendations are generated and stored. • The action is logged.
Alternative Flow	**If the user is Admin:** 1. Admin selects a Student. 2. Admin selects "Assessment History". 3. Admin selects a specific assessment. 4. Admin clicks "Generate Recommendations" based on configs. 5. System generates recommendations.
Assumptions	• The recommendation engine has appropriate configs. • The assessment results are valid and sufficient for analysis.
Exceptions	• The recommendation engine fails or times out. • The results provided are missing or corrupted.

4.9Use Case Name: Review Recommendation

Use Case ID	**UC-09**
Actors	**All users**

Description	Allows **students**, **admins**, and **tutors** to review recommendations for a specific **student**.
Preconditions	The **user** is logged in and has access to the UpsWing! platform
Basic Flow	**Student** 1. Student selects “Assessment History”. 2. Student selects a specific assessment. 3. System retrieves assessment details. 4. Student selects “Recommendations”. 5. System retrieves recommendations. 6. Students may review details.
Postconditions	• Recommendations have been retrieved. • The action is logged.
Alternative Flow	**If the user is Admin or Tutor:** 1a. The user selects a Student. 2a. Go to Basic Flow Step 1.
Assumptions	• Recommendations have been generated for the student.
Exceptions	• No recommendations are found for the student.

4.10 Use Case Name: Override Recommendation

Use Case ID	**UC-10**
Actors	**Admin**
Description	Allows **admins** to adjust or manually override automatically generated recommendations.
Preconditions	• The **admin** is logged in and has access to the UpsWing! Platform • The courses and lessons list is populated.
Basic Flow	1. Admin selects a Student. 2. Admin selects “Assessment History”. 3. Admin selects a specific assessment. 4. System retrieves assessment details. 5. Admin selects “Recommendations”. 6. Admin clicks “Override Recommendations”. 7. System sends a list of course/lessons. 8. Admin manually inputs recommended items or edits items. 9. System replaces the old recommendations with the manual override.

Postconditions	• Manual recommendations replace the automatic recommendations and are stored. • The action is logged.
Assumptions	• The admin has the pedagogical knowledge to make an appropriate manual recommendation.
Exceptions	• The manual items selected by the admin are invalid, non-existent, or incompatible. • The admin attempts to save an empty recommendation list.

4.11 Use Case Name: Review Real-Time Progress Report

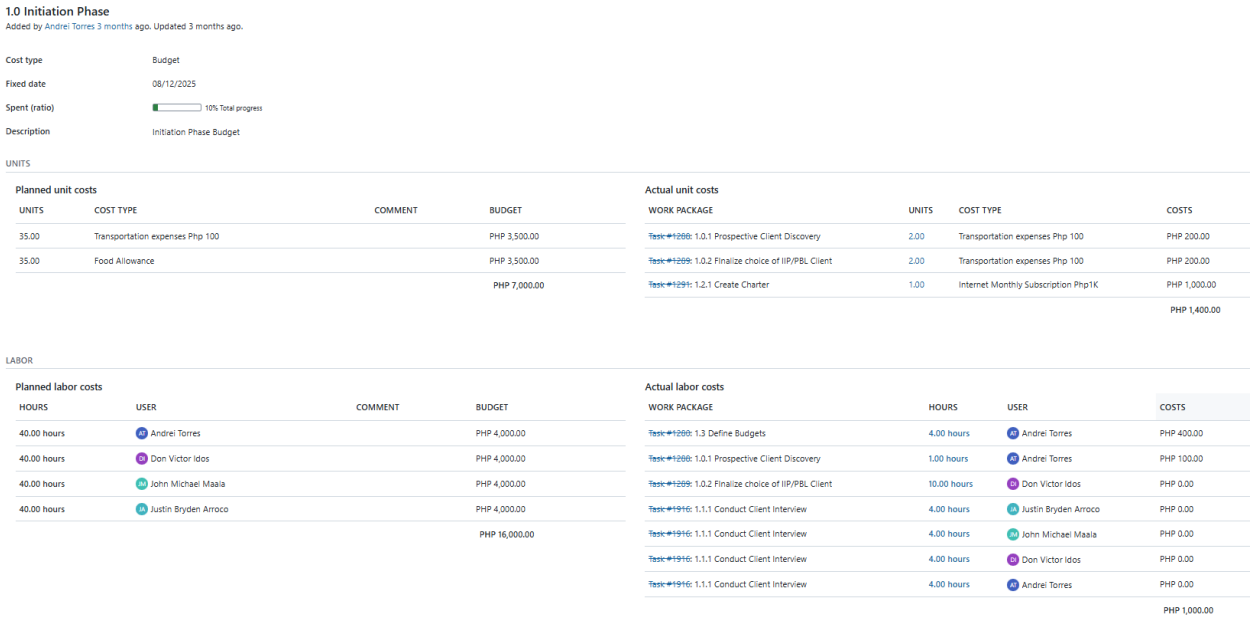
Use Case ID	**UC-11**
Actors	**All users**
Description	Allows **students**, **admins**, and **tutors** to view real-time progress report of a specific **student**.
Preconditions	The **user** is logged in and has access to the UpsWing! platform
Basic Flow	**Student** 1. Student selects “Progress Report”. 2. System retrieves data and generates analytics. 3. System sends the analytics and trends. 4. Student can view their progress report. 5. Student may apply filters to the report.
Postconditions	• The Progress Report page can be viewed and analytics have been generated. • The action is logged.
Alternative Flow	**If the user is Admin or Tutor:** 1a. The user selects a Student. 2a. Go to Basic Flow Step 1.
Assumptions	• The student has completed at least one assessment and has gotten a result. • The analytics engine is configured and running.
Exceptions	• Analytics generation fails or times out. • No data is found for the student.

4.12 Use Case Name: Export Progress Report

Use Case ID	**UC-12**

Actors	**All users**
Description	Allows **students**, **admins**, and **tutors** to export a real-time progress report of a specific **student**.
Preconditions	The **user** is logged in and has access to the UpsWing! platform
Basic Flow	**Student** 1. Student selects “Progress Report”. 2. System retrieves data and generates analytics. 3. System sends the analytics and trends. 4. Student may apply filters to the report. 5. Student clicks Export. 6. System generates file and downloads to local storage.
Postconditions	• Progress Report file is downloaded (CSV or PDF). • The action is logged.
Alternative Flow	**If the user is Admin or Tutor:** 1. The user selects a Student. 2. Go to Basic Flow Step 1.
Assumptions	• The student has completed at least one assessment and has gotten a result. • The analytics engine is configured and running.
Exceptions	• Analytics generation fails or times out. • No data is found for the student. • File generation fails. • File download fails.

FIGURE XV: OPENPROJECT BUDGETS 1.0



1.0 Initiation Phase

Added by Andrei Torres 3 months ago. Updated 3 months ago.

Cost type	Budget
Fixed date	08/12/2025
Spent (ratio)	10% Total progress
Description	Initiation Phase Budget

UNITS

Planned unit costs

UNITS	COST TYPE	COMMENT	BUDGET
35.00	Transportation expenses Php 100		PHP 3,500.00
35.00	Food Allowance		PHP 3,500.00
			PHP 7,000.00

Actual unit costs

WORK PACKAGE	UNITS	COST TYPE	COSTS
~~Task #1200~~: 1.0.1 Prospective Client Discovery	2.00	Transportation expenses Php 100	PHP 200.00
~~Task #1289~~: 1.0.2 Finalize choice of IIP/PBL Client	2.00	Transportation expenses Php 100	PHP 200.00
~~Task #1291~~: 1.2.1 Create Charter	1.00	Internet Monthly Subscription Php1K	PHP 1,000.00
			PHP 1,400.00

LABOR

Planned labor costs

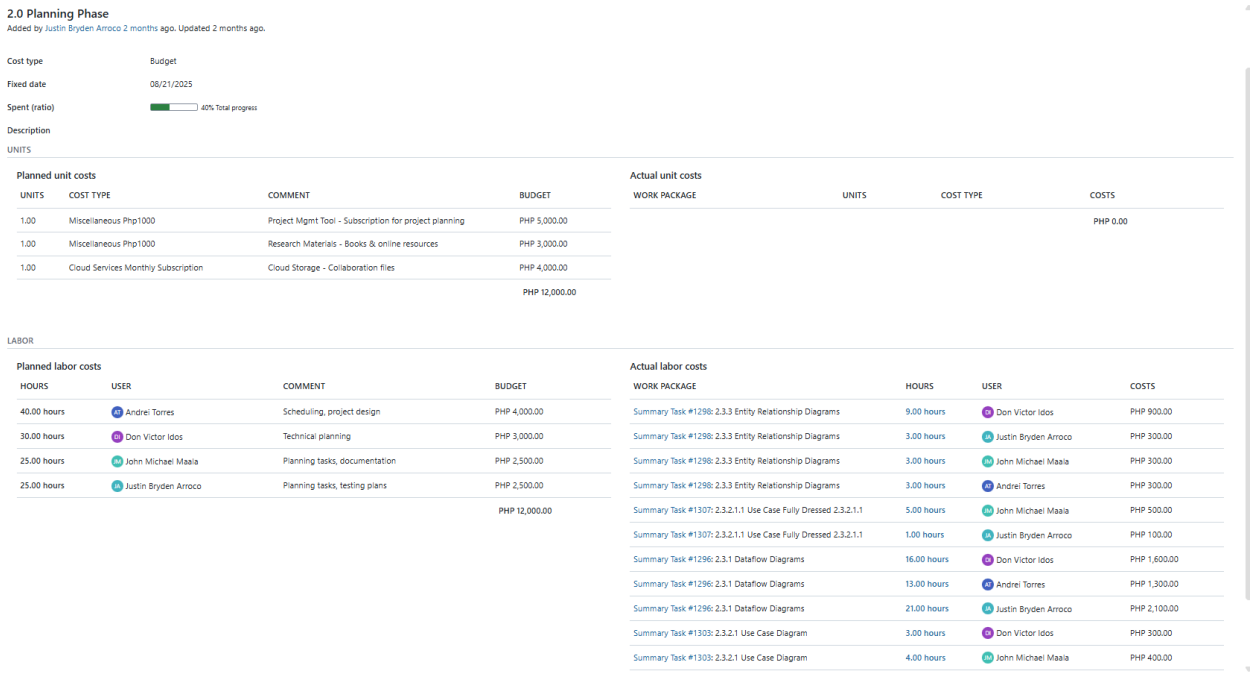
HOURS	USER	COMMENT	BUDGET
40.00 hours	Andrei Torres		PHP 4,000.00
40.00 hours	Don Victor Idos		PHP 4,000.00
40.00 hours	John Michael Maala		PHP 4,000.00
40.00 hours	Justin Bryden Arroco		PHP 4,000.00
			PHP 16,000.00

Actual labor costs

WORK PACKAGE	HOURS	USER	COSTS
~~Task #1200~~: 1.3 Define Budgets	4.00 hours	Andrei Torres	PHP 400.00
~~Task #1200~~: 1.0.1 Prospective Client Discovery	1.00 hours	Andrei Torres	PHP 100.00
~~Task #1289~~: 1.0.2 Finalize choice of IIP/PBL Client	10.00 hours	Don Victor Idos	PHP 0.00
~~Task #1916~~: 1.1.1 Conduct Client Interview	4.00 hours	Justin Bryden Arroco	PHP 0.00
~~Task #1916~~: 1.1.1 Conduct Client Interview	4.00 hours	John Michael Maala	PHP 0.00
~~Task #1916~~: 1.1.1 Conduct Client Interview	4.00 hours	Don Victor Idos	PHP 0.00
~~Task #1916~~: 1.1.1 Conduct Client Interview	4.00 hours	Andrei Torres	PHP 0.00
			PHP 1,000.00

|Link|

FIGURE XVI: OPENPROJECT BUDGETS 2.0



2.0 Planning Phase

Added by Justin Bryden Arroco 2 months ago. Updated 2 months ago.

Cost type	Budget
Fixed date	08/21/2025
Spent (ratio)	40% Total progress
Description	

UNITS

Planned unit costs

UNITS	COST TYPE	COMMENT	BUDGET
1.00	Miscellaneous Php1000	Project Mgmt Tool - Subscription for project planning	PHP 5,000.00
1.00	Miscellaneous Php1000	Research Materials - Books & online resources	PHP 3,000.00
1.00	Cloud Services Monthly Subscription	Cloud Storage - Collaboration files	PHP 4,000.00
			PHP 12,000.00

Actual unit costs

WORK PACKAGE	UNITS	COST TYPE	COSTS
			PHP 0.00

LABOR

Planned labor costs

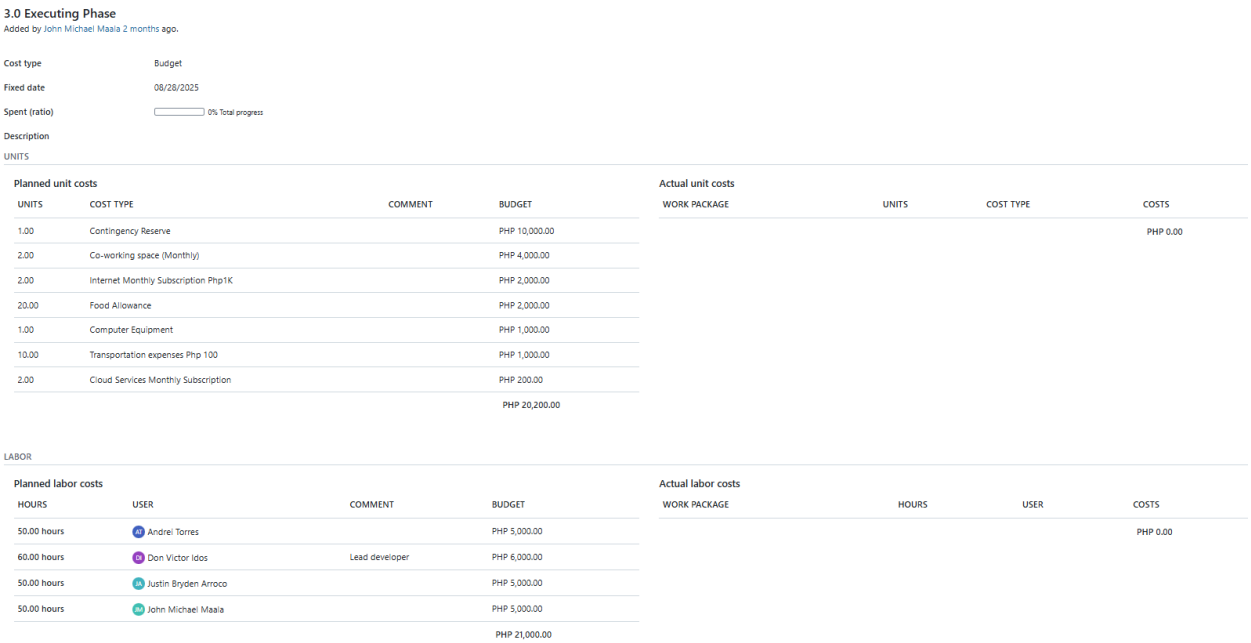
HOURS	USER	COMMENT	BUDGET
40.00 hours	Andrei Torres	Scheduling, project design	PHP 4,000.00
30.00 hours	Don Victor Idos	Technical planning	PHP 3,000.00
25.00 hours	John Michael Maala	Planning tasks, documentation	PHP 2,500.00
25.00 hours	Justin Bryden Arroco	Planning tasks, testing plans	PHP 2,500.00
			PHP 12,000.00

Actual labor costs

WORK PACKAGE	HOURS	USER	COSTS
Summary Task #1298: 2.3.3 Entity Relationship Diagrams	9.00 hours	Don Victor Idos	PHP 900.00
Summary Task #1298: 2.3.3 Entity Relationship Diagrams	3.00 hours	Justin Bryden Arroco	PHP 300.00
Summary Task #1298: 2.3.3 Entity Relationship Diagrams	3.00 hours	John Michael Maala	PHP 300.00
Summary Task #1298: 2.3.3 Entity Relationship Diagrams	3.00 hours	Andrei Torres	PHP 300.00
Summary Task #1307: 2.3.2.1.1 Use Case Fully Dressed 2.3.2.1.1	5.00 hours	John Michael Maala	PHP 500.00
Summary Task #1307: 2.3.2.1.1 Use Case Fully Dressed 2.3.2.1.1	1.00 hours	Justin Bryden Arroco	PHP 100.00
Summary Task #1296: 2.3.1 Dataflow Diagrams	16.00 hours	Don Victor Idos	PHP 1,600.00
Summary Task #1296: 2.3.1 Dataflow Diagrams	13.00 hours	Andrei Torres	PHP 1,300.00
Summary Task #1296: 2.3.1 Dataflow Diagrams	21.00 hours	Justin Bryden Arroco	PHP 2,100.00
Summary Task #1303: 2.3.2.1 Use Case Diagram	3.00 hours	Don Victor Idos	PHP 300.00
Summary Task #1303: 2.3.2.1 Use Case Diagram	4.00 hours	John Michael Maala	PHP 400.00

|Link|

FIGURE XVII: OPENPROJECT BUDGETS 3.0



3.0 Executing Phase
Added by John Michael Maala 2 months ago.

Cost type	Budget
Fixed date	08/28/2025
Spent (ratio)	0% Total progress
Description	

UNITS

Planned unit costs

UNITS	COST TYPE	COMMENT	BUDGET
1.00	Contingency Reserve		PHP 10,000.00
2.00	Co-working space (Monthly)		PHP 4,000.00
2.00	Internet Monthly Subscription Php1K		PHP 2,000.00
20.00	Food Allowance		PHP 2,000.00
1.00	Computer Equipment		PHP 1,000.00
10.00	Transportation expenses Php 100		PHP 1,000.00
2.00	Cloud Services Monthly Subscription		PHP 200.00
			PHP 20,200.00

Actual unit costs

WORK PACKAGE	UNITS	COST TYPE	COSTS
			PHP 0.00

LABOR

Planned labor costs

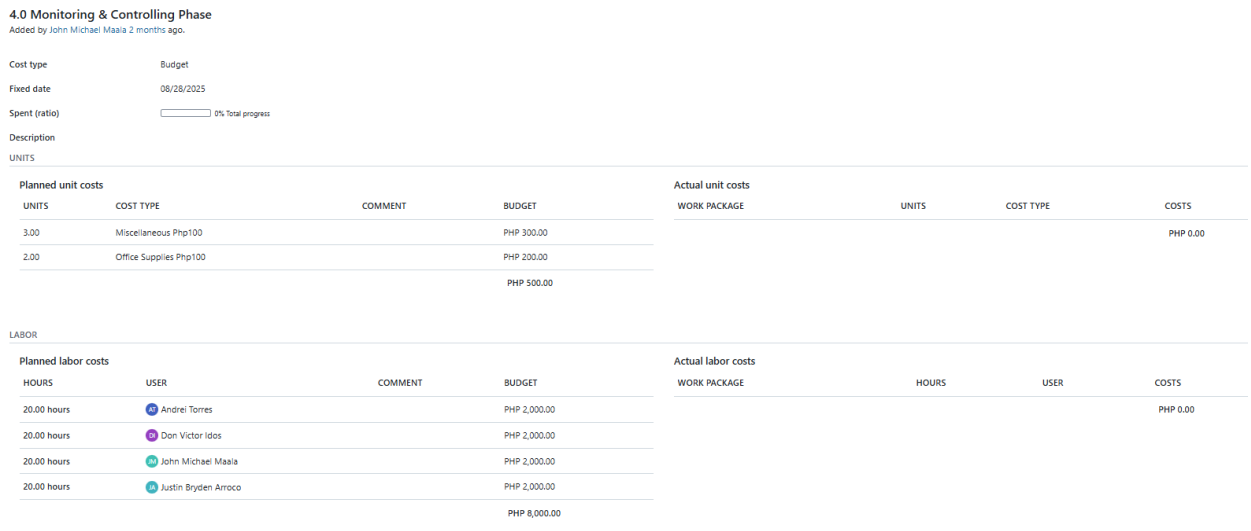
HOURS	USER	COMMENT	BUDGET
50.00 hours	Andrei Torres		PHP 5,000.00
60.00 hours	Don Victor Idos	Lead developer	PHP 6,000.00
50.00 hours	Justin Bryden Arroco		PHP 5,000.00
50.00 hours	John Michael Maala		PHP 5,000.00
			PHP 21,000.00

Actual labor costs

WORK PACKAGE	HOURS	USER	COSTS
			PHP 0.00

|Link|

FIGURE XVIII: OPENPROJECT BUDGETS 4.0



4.0 Monitoring & Controlling Phase
Added by John Michael Maala 2 months ago.

Cost type	Budget
Fixed date	08/28/2025
Spent (ratio)	0% Total progress
Description	

UNITS

Planned unit costs

UNITS	COST TYPE	COMMENT	BUDGET
3.00	Miscellaneous Php100		PHP 300.00
2.00	Office Supplies Php100		PHP 200.00
			PHP 500.00

Actual unit costs

WORK PACKAGE	UNITS	COST TYPE	COSTS
			PHP 0.00

LABOR

Planned labor costs

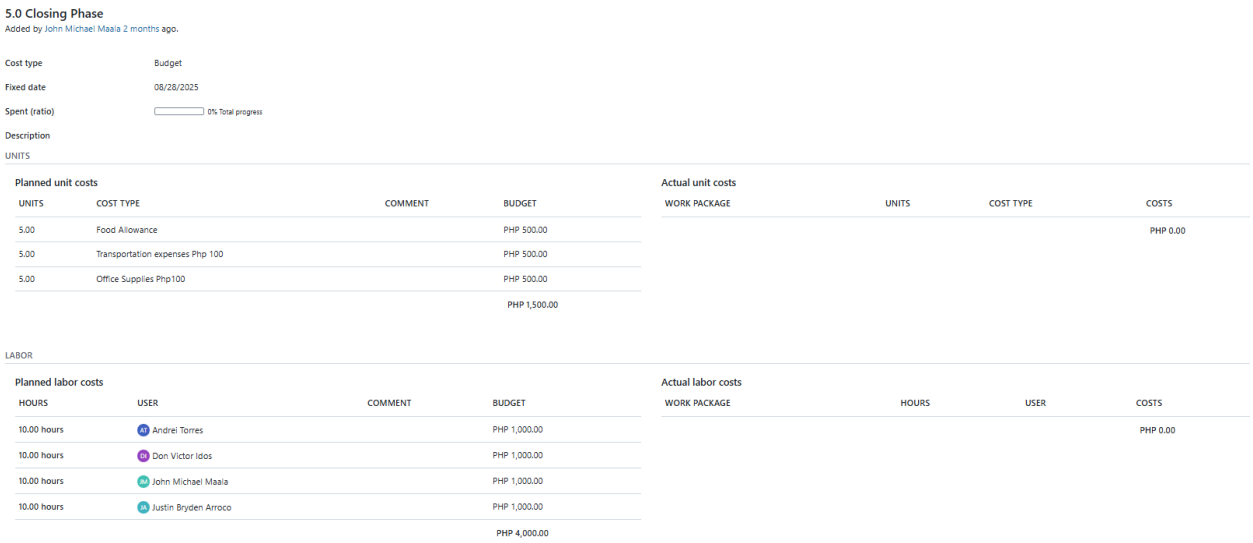
HOURS	USER	COMMENT	BUDGET
20.00 hours	Andrei Torres		PHP 2,000.00
20.00 hours	Don Victor Idos		PHP 2,000.00
20.00 hours	John Michael Maala		PHP 2,000.00
20.00 hours	Justin Bryden Arroco		PHP 2,000.00
			PHP 8,000.00

Actual labor costs

WORK PACKAGE	HOURS	USER	COSTS
			PHP 0.00

|Link|

FIGURE XIX: OPENPROJECT BUDGETS 5.0



5.0 Closing Phase

Added by John Michael Maala 2 months ago.

Cost type	Budget
Fixed date	08/28/2025
Spent (ratio)	0% Total progress
Description	

UNITS

Planned unit costs

UNITS	COST TYPE	COMMENT	BUDGET
5.00	Food Allowance		PHP 500.00
5.00	Transportation expenses Php 100		PHP 500.00
5.00	Office Supplies Php100		PHP 500.00
			PHP 1,500.00

Actual unit costs

WORK PACKAGE	UNITS	COST TYPE	COSTS
			PHP 0.00

LABOR

Planned labor costs

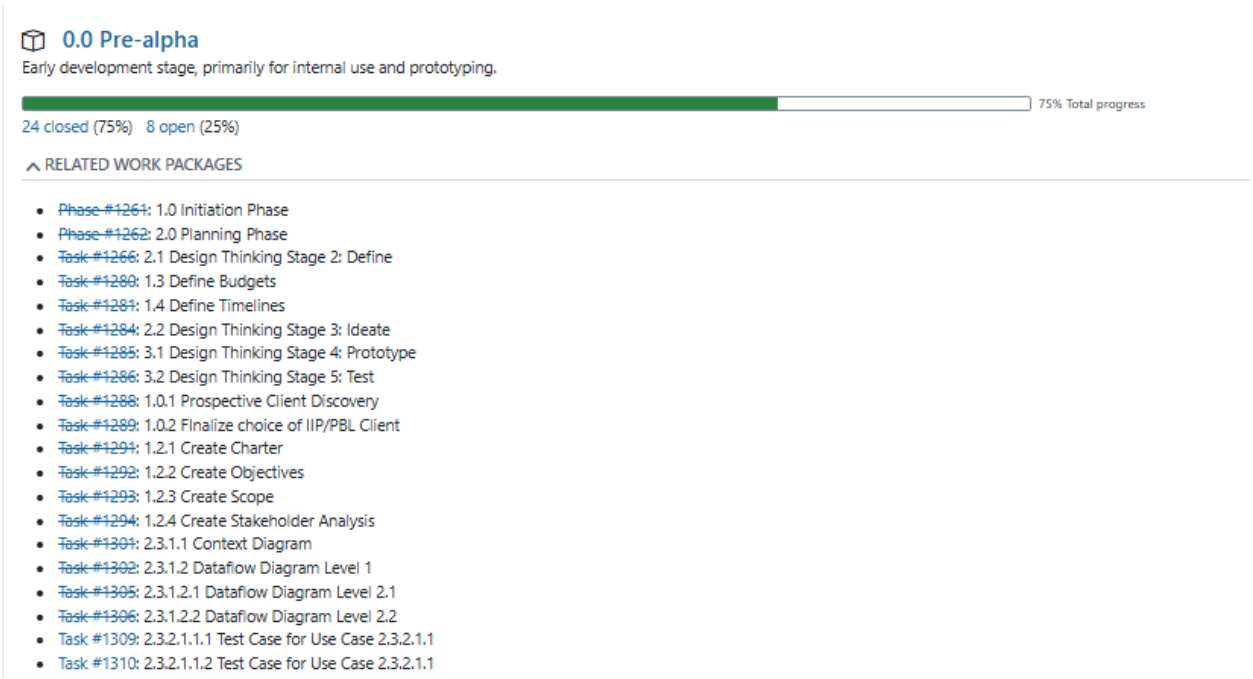
HOURS	USER	COMMENT	BUDGET
10.00 hours	Andrei Torres		PHP 1,000.00
10.00 hours	Don Victor Idos		PHP 1,000.00
10.00 hours	John Michael Maala		PHP 1,000.00
10.00 hours	Justin Bryden Arroco		PHP 1,000.00
			PHP 4,000.00

Actual labor costs

WORK PACKAGE	HOURS	USER	COSTS
			PHP 0.00

|Link|

FIGURE XX: OPENPROJECT ROADMAP 0.0



0.0 Pre-alpha

Early development stage, primarily for internal use and prototyping.

75% Total progress

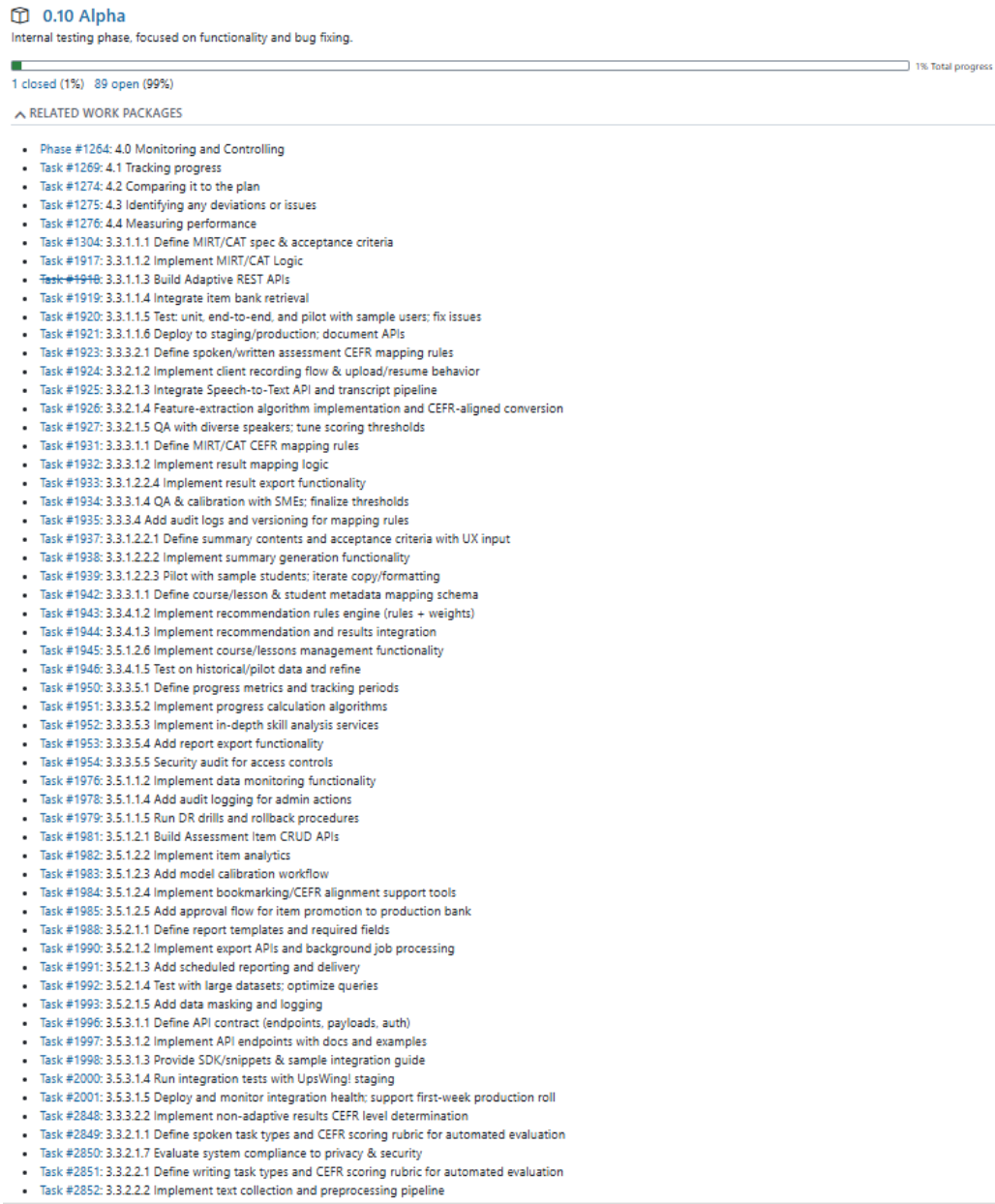
24 closed (75%) 8 open (25%)

RELATED WORK PACKAGES

- ~~Phase #1261~~: 1.0 Initiation Phase
- ~~Phase #1262~~: 2.0 Planning Phase
- ~~Task #1266~~: 2.1 Design Thinking Stage 2: Define
- ~~Task #1280~~: 1.3 Define Budgets
- ~~Task #1281~~: 1.4 Define Timelines
- ~~Task #1284~~: 2.2 Design Thinking Stage 3: Ideate
- ~~Task #1285~~: 3.1 Design Thinking Stage 4: Prototype
- ~~Task #1286~~: 3.2 Design Thinking Stage 5: Test
- ~~Task #1288~~: 1.0.1 Prospective Client Discovery
- ~~Task #1289~~: 1.0.2 Finalize choice of IIP/PBL Client
- ~~Task #1291~~: 1.2.1 Create Charter
- ~~Task #1292~~: 1.2.2 Create Objectives
- ~~Task #1293~~: 1.2.3 Create Scope
- ~~Task #1294~~: 1.2.4 Create Stakeholder Analysis
- ~~Task #1301~~: 2.3.1.1 Context Diagram
- ~~Task #1302~~: 2.3.1.2 Dataflow Diagram Level 1
- ~~Task #1305~~: 2.3.1.2.1 Dataflow Diagram Level 2.1
- ~~Task #1306~~: 2.3.1.2.2 Dataflow Diagram Level 2.2
- Task #1309: 2.3.2.1.1.1 Test Case for Use Case 2.3.2.1.1
- Task #1310: 2.3.2.1.1.2 Test Case for Use Case 2.3.2.1.1

|Link|

FIGURE XX: OPENPROJECT ROADMAP 0.10



0.10 Alpha

Internal testing phase, focused on functionality and bug fixing.

1% Total progress

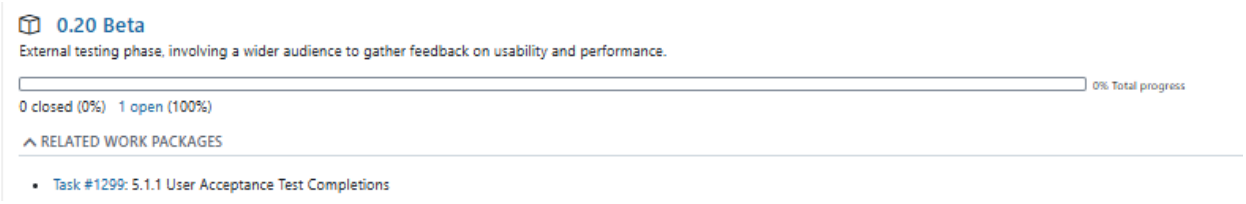
1 closed (1%) 89 open (99%)

RELATED WORK PACKAGES

- Phase #1264: 4.0 Monitoring and Controlling
- Task #1269: 4.1 Tracking progress
- Task #1274: 4.2 Comparing it to the plan
- Task #1275: 4.3 Identifying any deviations or issues
- Task #1276: 4.4 Measuring performance
- Task #1304: 3.3.1.1.1 Define MIRT/CAT spec & acceptance criteria
- Task #1917: 3.3.1.1.2 Implement MIRT/CAT Logic
- ~~Task #1918~~: 3.3.1.1.3 Build Adaptive REST APIs
- Task #1919: 3.3.1.1.4 Integrate item bank retrieval
- Task #1920: 3.3.1.1.5 Test: unit, end-to-end, and pilot with sample users; fix issues
- Task #1921: 3.3.1.1.6 Deploy to staging/production; document APIs
- Task #1923: 3.3.3.2.1 Define spoken/written assessment CEFR mapping rules
- Task #1924: 3.3.2.1.2 Implement client recording flow & upload/resume behavior
- Task #1925: 3.3.2.1.3 Integrate Speech-to-Text API and transcript pipeline
- Task #1926: 3.3.2.1.4 Feature-extraction algorithm implementation and CEFR-aligned conversion
- Task #1927: 3.3.2.1.5 QA with diverse speakers; tune scoring thresholds
- Task #1931: 3.3.3.1.1 Define MIRT/CAT CEFR mapping rules
- Task #1932: 3.3.3.1.2 Implement result mapping logic
- Task #1933: 3.3.1.2.2.4 Implement result export functionality
- Task #1934: 3.3.3.1.4 QA & calibration with SMEs; finalize thresholds
- Task #1935: 3.3.3.4 Add audit logs and versioning for mapping rules
- Task #1937: 3.3.1.2.2.1 Define summary contents and acceptance criteria with UX input
- Task #1938: 3.3.1.2.2.2 Implement summary generation functionality
- Task #1939: 3.3.1.2.2.3 Pilot with sample students; iterate copy/formatting
- Task #1942: 3.3.3.1.1 Define course/lesson & student metadata mapping schema
- Task #1943: 3.3.4.1.2 Implement recommendation rules engine (rules + weights)
- Task #1944: 3.3.4.1.3 Implement recommendation and results integration
- Task #1945: 3.5.1.2.6 Implement course/lessons management functionality
- Task #1946: 3.3.4.1.5 Test on historical/pilot data and refine
- Task #1950: 3.3.3.5.1 Define progress metrics and tracking periods
- Task #1951: 3.3.3.5.2 Implement progress calculation algorithms
- Task #1952: 3.3.3.5.3 Implement in-depth skill analysis services
- Task #1953: 3.3.3.5.4 Add report export functionality
- Task #1954: 3.3.3.5.5 Security audit for access controls
- Task #1976: 3.5.1.1.2 Implement data monitoring functionality
- Task #1978: 3.5.1.1.4 Add audit logging for admin actions
- Task #1979: 3.5.1.1.5 Run DR drills and rollback procedures
- Task #1981: 3.5.1.2.1 Build Assessment Item CRUD APIs
- Task #1982: 3.5.1.2.2 Implement item analytics
- Task #1983: 3.5.1.2.3 Add model calibration workflow
- Task #1984: 3.5.1.2.4 Implement bookmarking/CEFR alignment support tools
- Task #1985: 3.5.1.2.5 Add approval flow for item promotion to production bank
- Task #1988: 3.5.2.1.1 Define report templates and required fields
- Task #1990: 3.5.2.1.2 Implement export APIs and background job processing
- Task #1991: 3.5.2.1.3 Add scheduled reporting and delivery
- Task #1992: 3.5.2.1.4 Test with large datasets; optimize queries
- Task #1993: 3.5.2.1.5 Add data masking and logging
- Task #1996: 3.5.3.1.1 Define API contract (endpoints, payloads, auth)
- Task #1997: 3.5.3.1.2 Implement API endpoints with docs and examples
- Task #1998: 3.5.3.1.3 Provide SDK/snippets & sample integration guide
- Task #2000: 3.5.3.1.4 Run integration tests with UpsWing! staging
- Task #2001: 3.5.3.1.5 Deploy and monitor integration health; support first-week production roll
- Task #2848: 3.3.3.2.2 Implement non-adaptive results CEFR level determination
- Task #2849: 3.3.2.1.1 Define spoken task types and CEFR scoring rubric for automated evaluation
- Task #2850: 3.3.2.1.7 Evaluate system compliance to privacy & security
- Task #2851: 3.3.2.2.1 Define writing task types and CEFR scoring rubric for automated evaluation
- Task #2852: 3.3.2.2.2 Implement text collection and preprocessing pipeline

|Link|

FIGURE XXI: OPENPROJECT ROADMAP 0.20



|Link|

FIGURE XXII: OPENPROJECT WORK PACKAGES

ID	SUBJECT	TYPE	STATUS	ASSIGNEE	PRIORITY	VERSION
1261	1.0 Initiation Phase	PHASE	Closed	Andrei Torres	Normal	0.0 Pre-alpha
1280	1.3 Define Budgets	TASK	Closed	Andrei Torres	Normal	0.0 Pre-alpha
1281	1.4 Define Timelines	TASK	Closed	Andrei Torres	Normal	0.0 Pre-alpha
1282	1.2 PM Foundations Ch 2 Initiate a project	SUMMARY TASK	Closed	Andrei Torres	Normal	0.0 Pre-alpha
1291	1.2.1 Create Charter	TASK	Closed	John Michael Maala	Normal	0.0 Pre-alpha
1292	1.2.2 Create Objectives	TASK	Closed	Justin Bryden Arroco	Normal	0.0 Pre-alpha
1293	1.2.3 Create Scope	TASK	Closed	Justin Bryden Arroco	Normal	0.0 Pre-alpha
1294	1.2.4 Create Stakeholder Analysis	TASK	Closed	Don Victor Idos	Normal	0.0 Pre-alpha
1295	1.2.5 PM Foundations Ch 2 Initiate a project Completed	MILESTONE	Closed	Andrei Torres	Normal	0.0 Pre-alpha
1283	1.1 Design Thinking Stage 1: Empathize	TASK	Closed	Andrei Torres	Normal	-
1916	1.1.1 Conduct Client Interview	TASK	Closed	Don Victor Idos	Normal	-
1288	1.0.1 Prospective Client Discovery	TASK	Closed	Andrei Torres	Normal	0.0 Pre-alpha
1289	1.0.2 Finalize choice of IIP/PBL Client	TASK	Closed	Andrei Torres	Normal	0.0 Pre-alpha
1262	2.0 Planning Phase	PHASE	Closed	-	Normal	0.0 Pre-alpha
1287	2.3 Requirements and Analysis Design Diagrams	SUMMARY TASK	Closed	Andrei Torres	Normal	0.0 Pre-alpha
1296	2.3.1 Dataflow Diagrams	SUMMARY TASK	In progress	Justin Bryden Arroco	Normal	0.0 Pre-alpha
1301	2.3.1.1 Context Diagram	TASK	Closed	Justin Bryden Arroco	Normal	0.0 Pre-alpha
1302	2.3.1.2 Dataflow Diagram Level 1	TASK	Closed	Justin Bryden Arroco	Normal	0.0 Pre-alpha
1305	2.3.1.2.1 Dataflow Diagram Level 2.1	TASK	Closed	Justin Bryden Arroco	Normal	0.0 Pre-alpha
1306	2.3.1.2.2 Dataflow Diagram Level 2.2	TASK	Closed	Justin Bryden Arroco	Normal	0.0 Pre-alpha
1297	2.3.2 Use Cases	SUMMARY TASK	In progress	Justin Bryden Arroco	Normal	0.0 Pre-alpha
1303	2.3.2.1 Use Case Diagram	SUMMARY TASK	In progress	John Michael Maala	Normal	0.0 Pre-alpha
1307	2.3.2.1.1 Use Case Fully Dressed 2.3.2.1.1	SUMMARY TASK	New	Justin Bryden Arroco	Normal	0.0 Pre-alpha
1309	2.3.2.1.1.1 Test Case for Use Case 2.3.2.1.1	TASK	In progress	John Michael Maala	Normal	0.0 Pre-alpha
1310	2.3.2.1.1.2 Test Case for Use Case 2.3.2.1.1	TASK	In progress	John Michael Maala	Normal	0.0 Pre-alpha
1308	2.3.2.1.2 Use Case Fully Dressed 2.3.2.1.2	SUMMARY TASK	Closed	John Michael Maala	Normal	0.0 Pre-alpha
1298	2.3.3 Entity Relationship Diagrams	SUMMARY TASK	In progress	Don Victor Idos	Normal	0.0 Pre-alpha
3831	2.3.4 Activity Diagram with Swimlanes	SUMMARY TASK	New	John Michael Maala	Normal	-
1278	2.4 Allocate Resources	SUMMARY TASK	Closed	Andrei Torres	Normal	0.0 Pre-alpha

|Link|

FIGURE XXIII: OPENPROJECT WORK PACKAGES II

1279	2.5 Develop strategies for managing risks	SUMMARY TASK	In progress	AT	Andrei Torres	Normal	0.0 Pre-alpha
1267	2.6 Set deadlines	SUMMARY TASK	Closed	AT	Andrei Torres	Normal	0.0 Pre-alpha
1266	2.1 Design Thinking Stage 2: Define	TASK	Closed	AT	Andrei Torres	Normal	0.0 Pre-alpha
1284	2.2 Design Thinking Stage 3: Ideate	TASK	Closed	AT	Andrei Torres	Normal	0.0 Pre-alpha
1263	3.0 Executing Phase	PHASE	To be scheduled	-		Normal	-
1285	3.1 Design Thinking Stage 4: Prototype	TASK	Closed	DI	Don Victor Idos	Normal	0.0 Pre-alpha
1286	3.2 Design Thinking Stage 5: Test	TASK	Closed	DI	Don Victor Idos	Normal	0.0 Pre-alpha
1268	3.3 Student Assessment	EPIC	New	-		Normal	-
1290	3.3.1 Adaptive Assessment	USER STORY	New	-		Normal	0.10 Alpha
1300	3.3.1.1 Deliver Adaptive Assessment Service	FEATURE	New	-		Normal	0.10 Alpha
1304	3.3.1.1.1 Define MIRT/CAT spec & acceptance criteria	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1917	3.3.1.1.2 Implement MIRT/CAT Logic	TASK	In progress	DI	Don Victor Idos	Normal	0.10 Alpha
1918	3.3.1.1.3 Build Adaptive REST APIs	TASK	Closed	AT	Andrei Torres	Normal	0.10 Alpha
2842	3.3.1.1.3.1 Design and implement Adaptive Assessment Session Management	SUB TASK	New	DI	Don Victor Idos	Normal	-
2843	3.3.1.1.3.2 Design API request and response header/payload structure	SUB TASK	New	-		Normal	-
2844	3.3.1.1.3.3 Define API Endpoints for overall assessment flow	SUB TASK	New	-		Normal	-
1919	3.3.1.1.4 Integrate item bank retrieval	TASK	New	JM	John Michael Maala	Normal	0.10 Alpha
1920	3.3.1.1.5 Test: unit, end-to-end, and pilot with sample users; fix issues	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1921	3.3.1.1.6 Deploy to staging/production; document APIs	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
2845	3.3.2 Non-Adaptive Assessment	USER STORY	New	-		Normal	0.10 Alpha
1922	3.3.2.1 AI-Powered Speaking Assessment Service	FEATURE	New	-		Normal	0.10 Alpha
2849	3.3.2.1.1 Define spoken task types and CEFR scoring rubric for automated evaluation	TASK	New	-		Normal	0.10 Alpha
1924	3.3.2.1.2 Implement client recording flow & upload/resume behavior	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1925	3.3.2.1.3 Integrate Speech-to-Text API and transcript pipeline	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1926	3.3.2.1.4 Feature-extraction algorithm implementation and CEFR-aligned conversion	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1927	3.3.2.1.5 QA with diverse speakers; tune scoring thresholds	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
2861	3.3.2.1.6 Build Speaking Assessment REST APIs	TASK	New	-		Normal	0.10 Alpha
2850	3.3.2.1.7 Evaluate system compliance to privacy & security	TASK	New	-		Normal	0.10 Alpha
2847	3.3.2.2 AI-Powered Writing Assessment Service	FEATURE	New	-		Normal	0.10 Alpha

FIGURE XXIV: OPENPROJECT WORK PACKAGES III

2852	3.3.2.2.2 Implement text collection and preprocessing pipeline	TASK	New	-		Normal	0.10 Alpha
2853	3.3.2.2.3 Integrate NLP feature extraction for linguistic analysis	TASK	New	-		Normal	0.10 Alpha
2854	3.3.2.2.4 Implement LLM-based content and organization scoring	TASK	New	-		Normal	0.10 Alpha
2862	3.3.2.2.5 Build Writing Assessment REST APIs	TASK	New	-		Normal	0.10 Alpha
1929	3.3.3 Results & Summaries	USER STORY	New	-		Normal	0.10 Alpha
1930	3.3.3.1 Adaptive CEFR Mapping & Result Generation	FEATURE	New	-		Normal	0.10 Alpha
1931	3.3.3.1.1 Define MIRT/CAT CEFR mapping rules	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1932	3.3.3.1.2 Implement result mapping logic	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1934	3.3.3.1.4 QA & calibration with SMEs; finalize thresholds	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
2846	3.3.3.2 Non-Adaptive CEFR Mapping & Result Generation	FEATURE	New	-		Normal	0.10 Alpha
1923	3.3.3.2.1 Define spoken/written assessment CEFR mapping rules	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
2848	3.3.3.2.2 Implement non-adaptive results CEFR level determination	TASK	New	-		Normal	0.10 Alpha
2855	3.3.3.2.3 QA with diverse writing/speaking samples and expert validation	TASK	New	-		Normal	0.10 Alpha
1936	3.3.3.3 Student Result Summary & Visualization	FEATURE	New	-		Normal	0.10 Alpha
1937	3.3.1.2.2.1 Define summary contents and acceptance criteria with UX input	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1938	3.3.1.2.2.2 Implement summary generation functionality	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1939	3.3.1.2.2.3 Pilot with sample students; iterate copy/formatting	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1933	3.3.1.2.2.4 Implement result export functionality	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
2865	3.3.1.2.2.5 Implement in-depth result analysis services	TASK	New	-		Normal	0.10 Alpha
1935	3.3.3.4 Add audit logs and versioning for mapping rules	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1949	3.3.3.5 Student Progress Reports	FEATURE	New	-		Normal	0.10 Alpha
1950	3.3.3.5.1 Define progress metrics and tracking periods	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1951	3.3.3.5.2 Implement progress calculation algorithms	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1952	3.3.3.5.3 Implement in-depth skill analysis services	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1953	3.3.3.5.4 Add report export functionality	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
1954	3.3.3.5.5 Security audit for access controls	TASK	New	DI	Don Victor Idos	Normal	0.10 Alpha
2867	3.3.3.5.6 Build Progress Reports REST APIs	TASK	New	-		Normal	0.10 Alpha
2863	3.3.3.6 Build Results REST APIs	TASK	New	-		Normal	0.10 Alpha
1940	3.6 Personalized Learning Path	EPIC	New	-		Normal	-

FIGURE XXV: OPENPROJECT WORK PACKAGES IV

1941	3.3.4.1 Recommendation Engine for Learning Paths	USER STORY	New	-	Normal	0.10 Alpha
1942	3.3.3.1.1 Define course/lesson & student metadata mapping schema	TASK	New	Don Victor Idos	Normal	0.10 Alpha
1943	3.3.4.1.2 Implement recommendation rules engine (rules + weights)	TASK	New	Don Victor Idos	Normal	0.10 Alpha
1944	3.3.4.1.3 Implement recommendation and results integration	TASK	New	Don Victor Idos	Normal	0.10 Alpha
1946	3.3.4.1.5 Test on historical/pilot data and refine	TASK	New	Don Victor Idos	Normal	0.10 Alpha
2866	3.3.4.1.6 Build Recommendation REST APIs	TASK	New	-	Normal	0.10 Alpha
1947	3.4 Database Design & Implementation	EPIC	New	-	Normal	-
1948	3.4.1 Database Schema Design & Creation	USER STORY	New	-	Normal	0.10 Alpha
2870	3.4.1.1 Implement core assessment tables	TASK	New	-	Normal	-
2871	3.4.1.2 Implement question bank and configuration	TASK	New	-	Normal	0.10 Alpha
2872	3.4.1.3 Implement recommendation and course/lessons catalog tables	TASK	New	-	Normal	0.10 Alpha
2873	3.4.1.4 Optimize database performance	TASK	New	-	Normal	0.10 Alpha
2869	3.4.2 Database Integration & Testing	USER STORY	New	-	Normal	0.10 Alpha
1971	3.5 Admins	EPIC	New	-	Normal	-
1972	3.5.1 System Operations	USER STORY	New	-	Normal	0.10 Alpha
1973	3.5.1.1 Admin Monitoring Tools	FEATURE	New	-	Normal	0.10 Alpha
1976	3.5.1.1.2 Implement data monitoring functionality	TASK	New	Don Victor Idos	Normal	0.10 Alpha
1978	3.5.1.1.4 Add audit logging for admin actions	TASK	New	Don Victor Idos	Normal	0.10 Alpha
1979	3.5.1.1.5 Run DR drills and rollback procedures	TASK	New	Don Victor Idos	Normal	0.10 Alpha
1980	3.5.1.2 Content Management & Calibration Tools	FEATURE	New	-	Normal	0.10 Alpha
1981	3.5.1.2.1 Build Assessment Item CRUD APIs	TASK	New	Don Victor Idos	Normal	0.10 Alpha
1982	3.5.1.2.2 Implement item analytics	TASK	New	Don Victor Idos	Normal	0.10 Alpha
1983	3.5.1.2.3 Add model calibration workflow	TASK	New	Don Victor Idos	Normal	0.10 Alpha
1984	3.5.1.2.4 Implement bookmarking/CEFR alignment support tools	TASK	New	Don Victor Idos	Normal	0.10 Alpha
1985	3.5.1.2.5 Add approval flow for item promotion to production bank	TASK	New	Don Victor Idos	Normal	0.10 Alpha
1945	3.5.1.2.6 Implement course/lessons management functionality	TASK	New	Don Victor Idos	Normal	0.10 Alpha
2860	3.3.4.1.4.1 Implement CRUD course/lesson functionality	SUB TASK	New	-	Normal	-
2859	3.3.4.1.4.2 Implement course/lesson item metadata tuning	SUB TASK	New	-	Normal	-
2868	3.5.1.2.7 Assessment configuration management	TASK	New	-	Normal	0.10 Alpha

FIGURE XXV: OPENPROJECT WORK PACKAGES V

2864	3.5.1.3.2 Implement manual recommendation evaluation and adjustments	TASK	New	-	Normal	-
1959	3.5.1.3.3 Add marking functionality for validated assessment/recommendation outputs	TASK	New	DI Don Victor Idos	Normal	-
1986	3.5.2 Reporting	USER STORY	New	-	Normal	0.10 Alpha
1987	3.5.2.1 Reporting & Data Export Tooling	FEATURE	New	-	Normal	0.10 Alpha
1988	3.5.2.1.1 Define report templates and required fields	TASK	New	DI Don Victor Idos	Normal	0.10 Alpha
1990	3.5.2.1.2 Implement export APIs and background job processing	TASK	New	DI Don Victor Idos	Normal	0.10 Alpha
1991	3.5.2.1.3 Add scheduled reporting and delivery	TASK	New	DI Don Victor Idos	Normal	0.10 Alpha
1992	3.5.2.1.4 Test with large datasets; optimize queries	TASK	New	DI Don Victor Idos	Normal	0.10 Alpha
1993	3.5.2.1.5 Add data masking and logging	TASK	New	DI Don Victor Idos	Normal	0.10 Alpha
1994	3.5.3 Integration	USER STORY	New	-	Normal	0.10 Alpha
1995	3.5.3.1 API Integration	FEATURE	New	-	Normal	0.10 Alpha
1996	3.5.3.1.1 Define API contract (endpoints, payloads, auth)	TASK	New	DI Don Victor Idos	Normal	0.10 Alpha
1997	3.5.3.1.2 Implement API endpoints with docs and examples	TASK	New	DI Don Victor Idos	Normal	0.10 Alpha
1998	3.5.3.1.3 Provide SDK/snippets & sample integration guide	TASK	New	DI Don Victor Idos	Normal	0.10 Alpha
2000	3.5.3.1.4 Run integration tests with UpsWing! staging	TASK	New	DI Don Victor Idos	Normal	0.10 Alpha
2001	3.5.3.1.5 Deploy and monitor integration health: support first-week production roll	TASK	New	DI Don Victor Idos	Normal	0.10 Alpha
1264	4.0 Monitoring and Controlling	PHASE	To be scheduled	-	Normal	0.10 Alpha
1269	4.1 Tracking progress	TASK	New	-	Normal	0.10 Alpha
1274	4.2 Comparing it to the plan	TASK	New	-	Normal	0.10 Alpha
1275	4.3 Identifying any deviations or issues	TASK	New	-	Normal	0.10 Alpha
1276	4.4 Measuring performance	TASK	New	-	Normal	0.10 Alpha
1265	5.0 Closing Phase	PHASE	To be scheduled	-	Normal	-
1270	5.1 Completing all tasks	SUMMARY TASK	New	-	Normal	-
1299	5.1.1 User Acceptance Test Completions	TASK	New	-	Normal	0.20 Beta
1271	5.2 Obtaining final approvals	SUMMARY TASK	New	-	Normal	-
1272	5.3 Archiving project documentation	SUMMARY TASK	New	-	Normal	-
1273	5.4 Conducting a final review	SUMMARY TASK	New	-	Normal	-
1277	5.5 Project Turnover	SUMMARY TASK	New	-	Normal	-

FIGURE XXV: OPENPROJECT BACKLOGS

0.0 Pre-alpha 0

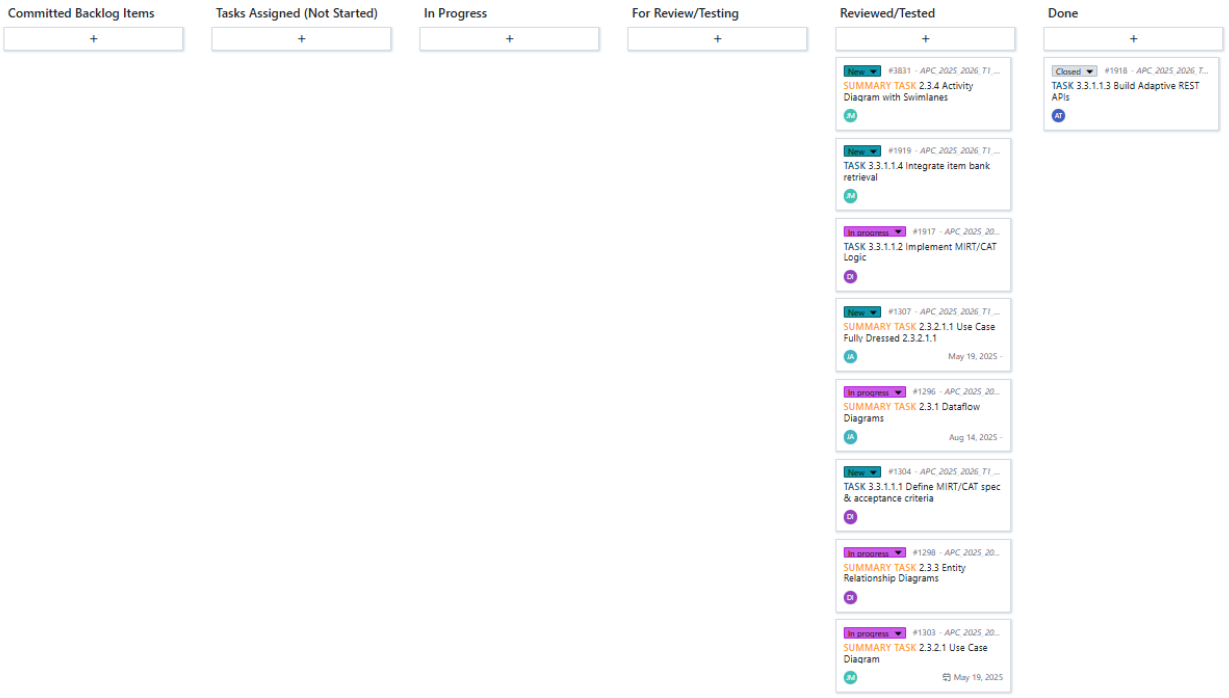
0.10 Alpha 0

1290	User story: 3.3.1 Adaptive Assessment	New
1300	Feature: 3.3.1.1 Deliver Adaptive Assessment Service	New
2845	User story: 3.3.2 Non-Adaptive Assessment	New
1930	Feature: 3.3.3.1 Adaptive CEFR Mapping & Result Generation	New
2846	Feature: 3.3.3.2 Non-Adaptive CEFR Mapping & Result Generation	New
1936	Feature: 3.3.3.3 Student Result Summary & Visualization	New
1949	Feature: 3.3.3.5 Student Progress Reports	New
1922	Feature: 3.3.2.1 AI-Powered Speaking Assessment Service	New
2847	Feature: 3.3.2.2 AI-Powered Writing Assessment Service	New
1929	User story: 3.3.3 Results & Summaries	New
1948	User story: 3.4.1 Database Schema Design & Creation	New
2869	User story: 3.4.2 Database Integration & Testing	New
1941	User story: 3.3.4.1 Recommendation Engine for Learning Paths	New
1972	User story: 3.5.1 System Operations	New
1973	Feature: 3.5.1.1 Admin Monitoring Tools	New
1980	Feature: 3.5.1.2 Content Management & Calibration Tools	New
1986	User story: 3.5.2 Reporting	New
1987	Feature: 3.5.2.1 Reporting & Data Export Tooling	New
1994	User story: 3.5.3 Integration	New
1995	Feature: 3.5.3.1 API Integration	New

0.20 Beta 0

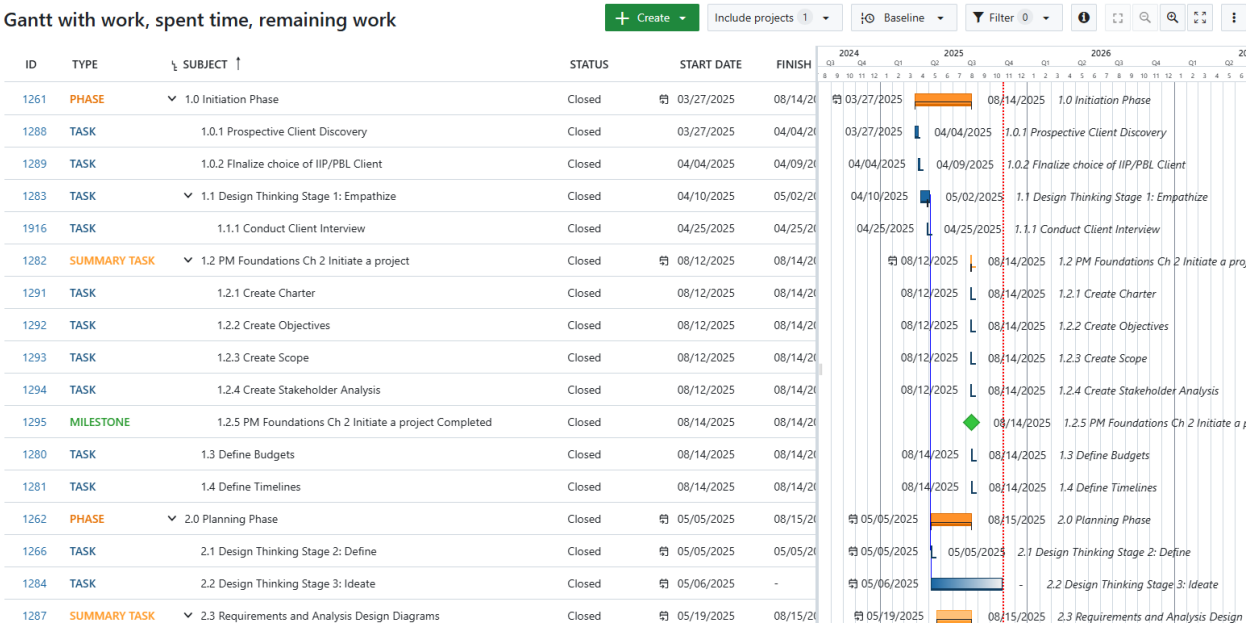
|Link|

FIGURE XXVI: OPENPROJECT BOARD



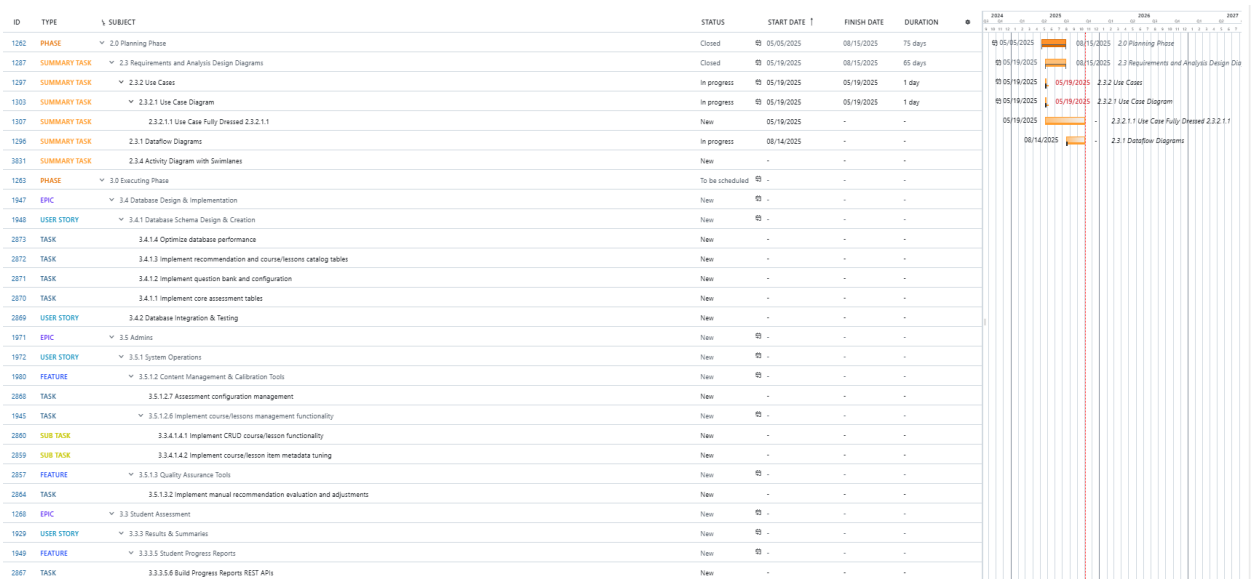
|Link|

FIGURE XXVII: OPENPROJECT GANTT CHART I



|Link|

FIGURE XXVII: OPENPROJECT GANTT CHART II



ID	TYPE	SUBJECT	STATUS	START DATE	FINISH DATE	DURATION
1262	PHASE	2.0 Planning Phase	Closed	05/05/2025	08/15/2025	75 days
1287	SUMMARY TASK	2.3 Requirements and Analysis Design Diagrams	Closed	05/19/2025	08/15/2025	65 days
1297	SUMMARY TASK	2.3.2 Use Cases	In progress	05/19/2025	05/19/2025	1 day
1303	SUMMARY TASK	2.3.2.1 Use Case Diagram	In progress	05/19/2025	05/19/2025	1 day
1307	SUMMARY TASK	2.3.2.1.1 Use Case Fully Dressed 2.3.2.1.1	New	05/19/2025	-	-
1296	SUMMARY TASK	2.3.1 Dataflow Diagrams	In progress	08/14/2025	-	-
3831	SUMMARY TASK	2.3.4 Activity Diagram with Swimlanes	New	-	-	-
1263	PHASE	3.0 Executing Phase	To be scheduled	-	-	-
1947	EPIC	3.4 Database Design & Implementation	New	-	-	-
1948	USER STORY	3.4.1 Database Schema Design & Creation	New	-	-	-
2873	TASK	3.4.1.4 Optimize database performance	New	-	-	-
2872	TASK	3.4.1.3 Implement recommendation and course/lessons catalog tables	New	-	-	-
2871	TASK	3.4.1.2 Implement question bank and configuration	New	-	-	-
2870	TASK	3.4.1.1 Implement core assessment tables	New	-	-	-
2869	USER STORY	3.4.2 Database Integration & Testing	New	-	-	-
1971	EPIC	3.5 Admins	New	-	-	-
1972	USER STORY	3.5.1 System Operations	New	-	-	-
1980	FEATURE	3.5.1.2 Content Management & Calibration Tools	New	-	-	-
2868	TASK	3.5.1.2.7 Assessment configuration management	New	-	-	-
1945	TASK	3.5.1.2.6 Implement course/lessons management functionality	New	-	-	-
2860	SUB TASK	3.3.4.1.4.1 Implement CRUD course/lesson functionality	New	-	-	-
2859	SUB TASK	3.3.4.1.4.2 Implement course/lesson item metadata tuning	New	-	-	-
2857	FEATURE	3.5.1.3 Quality Assurance Tools	New	-	-	-
2864	TASK	3.5.1.3.2 Implement manual recommendation evaluation and adjustments	New	-	-	-
1268	EPIC	3.3 Student Assessment	New	-	-	-
1929	USER STORY	3.3.3 Results & Summaries	New	-	-	-
1949	FEATURE	3.3.3.5 Student Progress Reports	New	-	-	-
2867	TASK	3.3.3.5.6 Build Progress Reports REST APIs	New	-	-	-

[Openproject Team Site URL Link]

[Github APC-SoCIT project repository URL Link]