Mobile Applications Development

AY2025 Semester 2

SCHOOL OF INFOCOMM TECHNOLOGY

Diploma in Data Science
Diploma in Information Technology

ASSIGNMENT

Individual/Team : Team of 4

Format : Stage 1 (Interim) (30%)
Stage 2 (Final) (40%)

Development Cut-Off Date/Time

Stage 1 (Interim) : **8 Dec 2025, 0900 hrs (Start of Week 8)**

Stage 2 (Final) : **9 Feb 2025, 0900 hrs (Start of Week17)**

Penalty for late submission:
10 marks per day (including Sunday and public holiday)

There is a total of 24 pages (including this page).

WARNING

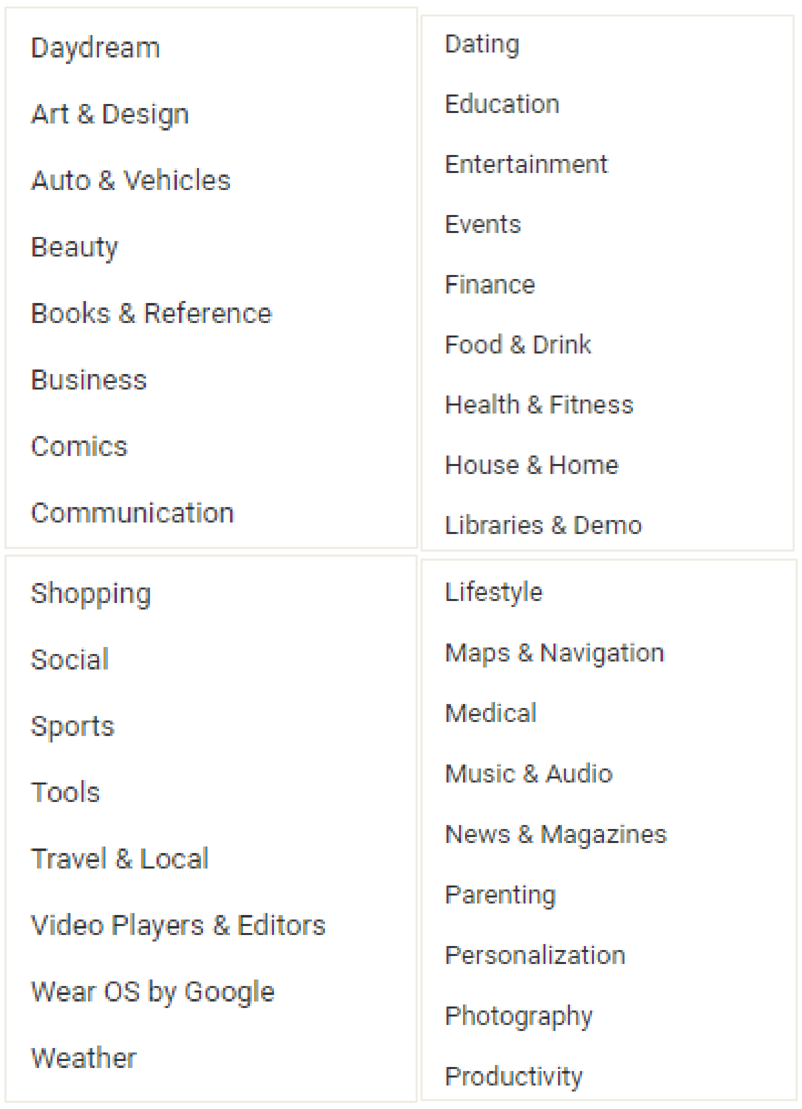
If a student is found to have submitted work not done by him/her, he/she will not be awarded any marks for this assignment. Disciplinary action will also be taken.
Similar action will be taken for the student who allows other student(s) to copy his/her work.

1. OBJECTIVES

- Demonstrate ability to implement a mobile app.
- Make conscientious consideration when designing a mobile app.
- Identify learning needs based on selected features, engage in learning process and implement the features.
- Experience publishing app to Google Play Store, receiving and managing user's ratings and comments via simulated environment.

2. INTRODUCTION

You are required to implement an android app of your choice while applying knowledge learnt in this module. Your application needs to fall under one of the categories listed in Figure 2.1.



Daydream
Art & Design
Auto & Vehicles
Beauty
Books & Reference
Business
Comics
Communication

Dating
Education
Entertainment
Events
Finance
Food & Drink
Health & Fitness
House & Home
Libraries & Demo

Shopping
Social
Sports
Tools
Travel & Local
Video Players & Editors
Wear OS by Google
Weather

Lifestyle
Maps & Navigation
Medical
Music & Audio
News & Magazines
Parenting
Personalization
Photography
Productivity

Figure 2.1 App Category

3. REQUIREMENTS

The following are the requirements necessary for the assignment to be credited accordingly.

3.1. General Requirements

All development code shall be committed to a team repository with the following repository requirements for account purposes:

Repository:

naming:
- MAD25_<TutorialGroup>_<TeamName>
- i.e. MAD25_T01_Team1

setting:
- Repository visibility to be ***Public***
- Readme.md shall be ***Enabled***
- License for the development shall be ***MIT license***

readme.md
- The following comment shall be added to the readme file as a disclaimer.

  ##Disclaimer
  This is a student assignment project for the Kotlin App Development module at Ngee Ann Polytechnic. Developed for educational purposes.

Collaborators:
- All team members are to be added as collaborators.
- A readme.md shall be created to indicate team member names to their respective Git username for grading and accounting purposes. **Failure to comply shall result in no grade credited to the team.**
- Add the following user account as a collaborator to grant access: lowkh

Language composition:
- The development for the app shall be in Kotlin or a hybrid of Kotlin code of no less than 80% of the development.
- This shall be based on Git measure of language as per provided by the repository measurement.

3.2. INTERIM PRESENTATION REQUIREMENTS

3.2.1. The team need to implement **<u>X number features</u>** of the following concepts during stage 1 of the development.

- Preliminary User Interface (UI) in line with App features
- Multimedia (Picture, Video, audio)
- Lazy Lists (Column, Row, Vertical Grid)
- Data Persistency (SharedPreferences/ SQLite/ Room/ Datastore)
- Other core topics listed on Android Developer Guides (https://developer.android.com/guide)

3.2.2. Create and include the following in the GitHub readme file preferably by Week 5.

- Introduction, motivation/objective, app category of the app, declaration of LLM used to assist in development if any.
- Task(s) and feature(s) allocation of each member in the team for Stage 1.
- Planned task(s) and feature(s) allocation of each member in the team for Stage 2.

3.2.3. A forum will be created in Brightspace to enable ratings and comments from your classmates during both presentations.

3.2.4. Each team will present to your tutor and your classmates and receive ratings and comments from your classmates via forum posts.

3.2.5. These ratings and comments will be taken into consideration for your interim presentation but would not directly affect your interim grade.

3.3. FINAL PRESENTATON REQUIREMENTS

If the Google Play Store is available, thou unlikely.
Else requirement 3.3.1 shall not be considered as part of the assignment.

3.3.1. Submit application to Google Play Store at the beginning of **Week 11**, fix any issues raised by Google and publish your application latest by **Week 15/16**. Take note that it takes a while for Google to approve the production release. Once the listing is live, edit the initial post in Padlet created on **Week 4** to include the URL of your app in Google Play Store. The team is also required to update the changes made in task(s) and feature(s) in the GitHub readme accordingly.

3.3.2. Each team member is to implement **at least 1 NEW feature** for your app with the following criteria:

- The feature requires creation of at least one new Activity or Service.
- The feature **must not** be similar or dependent on other team members. ***i.e. Stand-alone features.***
- Reasons for not meeting due to other features/codes/libraries not available will not be entertained. *i.e. Team member rely on another team member who did not deliver the library APIs included.*
- The feature requires you to implement a topic that would demonstrate self-learning, extension of concepts taught in the MAD module syllabus.

NOTE:
The fragments that are initially loaded in a *FragmentActivity* are **collectively** considered as 1 activity. Doing 5 fragments still counts as a single concept – Fragments.

3.3.3. Feedback from Stage 1 for changes may be considered as new feature for Stage 2 **IF** it meets the following criteria:

- The change would require more than 75% revamp of existing source code.
- The change would upgrade the existing feature to a new enhanced feature.
- The existing feature would be a showstopper for the app in totality. i.e. App would not be viable if the existing feature is not addressed.

3.3.4. The new feature **should not change or modify** the overall proposition of the app submitted in Stage 1.

NOTE:
Significant deviation from the original proposition of the application will cause **severe penalties** that will compromise the overall assignment grade for a student.

3.3.5. Each team will present to everyone in Week 17.

Factors to consider,

- Choosing appropriate API to implement the features.
- Designing appropriate user experience and user interface.
- Optimizing algorithm and memory usage.
- Unique feature of your app.
- Ease of using your app that can be quantified and not qualified.
- Use of LLM to get basic templates or boiler and its derivations from team members if any were used would need to be declared.

NOTE:
Unfamiliarity with any part of the program will cause penalties that will compromise the overall assignment grade for a student or team.

4. ACTIVITY PLAN

Suggestions on how to get started:

- **Brainstorm ideas**, research, and decide on the application to be implemented.
- **Discuss** with your tutor your idea.
- **Work out the user experience** needed for the system. **(Storyboard)**
- **Decide on appropriate sensors/services API**, layout and touch events required.
- **Divide the responsibilities** between you and your team-mates, making sure that the individual's tasks are clear, documented and agreed upon **upfront**. Remember to log the tasks clearly in the shared Git repository for the team.
- **Work out the main logic** of the application using MVC design pattern and modular programming techniques. Design each class carefully. Use functions appropriately. Think carefully about the return type and the parameters of each function.
- **Divide the programming task** into smaller stages. Work iteratively, ensuring that one stage works as desired before moving onto the next one.
- **Devise comprehensive test plan** and test the application. This is to prevent end users from downloading a defective or unworkable app.
- **Allocate at least 1 Week clearance period for submission to Google Play Store if It is available.** Fix issues highlighted by Google, fine tune for production release. It takes a while for Google to approve the production release, the team has been warned.

If the Google Play Store is applicable, all students are to note the following:

This module uses an official account provided to facilitate the teaching and learning of application development.
There are policies and rules governing the use of the Google Play Store.

Failure to comply may cause Google Play Store to terminate the account given to the team, resulting in heavy penalties to the **ENTIRE** team.

The team has been advised and assumed to have gone through the following for further information.

- Terms and Conditions from official Google Play Store.
- Quick Guide to Google Play Developer Policies.

5. DELIVERABLES

Assignment deliverables are segmented into **TWO** stages/ sessions:

Stage 1:

Due date of Stage 1 is as per dictated on cover page. The team is to take note.
All team members are required to merge changes into the main repository and commit/ submit to your project via Git and share the repository with your tutor.

Note: Changes **after** the deadline will be considered as Stage 2 contributions. All considerations will ONLY be based on the main repository branch. Alternative branches will NOT be considered.

The following information should be included in the Git readme file:

- Team members' names, student IDs
- Description of application. i.e. Brief explanation and the features to expect.
- Roles and contributions of each member for Stage 1 and proposed Stage 2.
- All relevant appendices if applicable (diagrams, screenshots, user guides)

Students should submit for Google Play approval as advised.

Google Play Store listing is driven by popularity, relevancy, and attractiveness of the icons and description. To simulate consumers' rating and ranking in Google Play Store, your classmates and tutor will rate and feedback about your app via Brightspace.

Each team will be given at least 30 minutes for presentation:
The following are **suggested** topics to cover

- Introduction of application
  - A brief description of the application, its functionalities, and reasons for implementing them.
- Description and explanation of
  - how related concepts are applied.
  - implementation problems encountered if any.
- Demonstration of your mobile app as of cut-off timing.
- Q&A session.

Stage 2:

Due date of Stage 2 is as per dictated on cover page. The team is to take note.
All team members are required to merge changes into the main repository and commit/submit to your project via Git and share the repository with your tutor.

Note: Changes **after** the deadline will be considered as late submissions.

The following information should be included in the Git readme file:

- Team members' names, student IDs
- Description of app and its current state of deployment.
- Roles and contributions of each member for Stage 1 and each member Stage 2 statuses.
- All relevant appendices if any (diagrams, screenshots, user guides)

Each team will be given at least 30 minutes for presentation:

- Introduction of application current state.
  - A brief description of the application then and now, the enhanced/ new functionalities and reasons for implementing them.
- Description and explanation of how NEW concepts are applied.
  - Accounting of tasks done by each individual member in reference to the new concepts implemented.
  - Implementation problems encountered, and the solutions implemented.
- Demonstration of your mobile app as of cut-off timing.
- Q&A session

6. MARKING SCHEME

Stage 1 (30%)

Component: Design Thinking and Ideation (Group 20%)

	Declared with LLM Assistance	**Declared without LLM Assistance**
Basic (0 ~ 1) “Stating as is”	• Implemented 1-2 concepts successfully. • Used LLM code without modifications or understanding. • Copy-pasted LLM personas and ideas without verification. • App theme consists of 1-3 generic ideas (social media, to-do list) and cannot explain value to problem. • No prototype or only basic sketches. • Code compiles but has issues or team cannot explain why code works. • No user research or testing conducted. • Generic LLM code present with no customization.	• Implemented 1-2 concepts successfully • Copy-pasted code from tutorials without understanding. • No user research or personas and generic assumptions are used. • App theme consists of 1-3 generic ideas and there is no evidence of ideation. • No prototype or only basic sketches. • Code compiles but has issues. • No comments, documentation or testing. • Cannot explain code or design choices.
Average (1 ~ 2) “State and explain”	• Implemented 3 concepts successfully. • Made minor customizations to LLM code and explains what it does. • Conducted basic user research with 1-2 pax and used LLM to organize findings. • Created basic personas with LLM assistance and identify some user problems. • Generated 4-6 ideas with LLM brainstorming and shows some variety. • Selected ideas with basic justification. • Created basic wireframes for 3-5 screens with one iteration based on peer feedback. • Identified and fixed 1-2 issues in LLM code generated. • Code works with minor issues. • Added basic comments and explanations.	• Implemented 3 concepts successfully • Explains code functionality and relationships • Conducted basic user research with 1-2 pax and created basic personas. • Identify 2-3 user problems with reasoning. • Generated 4-6 ideas through brainstorming and show some variety. • Selected ideas based on user needs. • Created wireframes for 3-5 screens with one iteration based on feedback. • Made minor improvements through refactoring. • Added meaningful comments explaining logic. • Applied basic error handling. • Code works with minor issues.

Intermediate (2 ~ 3) “State, explain and analyse”	• Implemented 3-5 concepts successfully with significant customization. • Significantly refactored LLM code and explains why structure is chosen. • Conducted structured user research of 4-6 pax and created detailed personas. • Used LLM to analyze interview data and identify patterns. • Identified 3-5 user problems with supporting evidence • Analyzed competitors (3-4 apps) and understand competitive landscape. • Generated 4-6 ideas using LLM-prompted techniques with good variety. • Evaluated ideas against user needs with clear ranking criteria. • Created detailed wireframes for 6-8 screens with complete user flows. • Conducted user testing with 4-6 pax and made 2-3 design iterations. • Fixed multiple issues in LLM code and tested thoroughly. • Combined LLM code with original implementations effectively. • Code works well for all main features and edge cases.	• Implemented 3-5 concepts successfully • Made multiple improvements through systematic refactoring. • Conducted structured user research of 4-6 pax and created detailed personas. • Identified 3-5 user problems with clear supporting evidence. • Analyzed competitors (3-4 apps) and understands competitive landscape. • Generated 4-6 ideas through brainstorming with good variety and originality. • Evaluated ideas systematically against user needs • Ranked ideas using clear selection criteria • Created detailed wireframes for 6-8 screens with complete user flows • Conducted user testing with 4-6 pax and made 2-3 design iterations. • Applied comprehensive error-handling for multiple scenarios. • Added meaningful documentation and comments • Code works well for all features and most edge cases.

Advanced (3 ~ 4) "State, explain, analyse and propose"	• Implemented 5+ concepts successfully with sophisticated customization. • Extensively refactored LLM code and created novel solutions beyond LLM capability. • Filled gaps where LLM knowledge was incomplete with own research. • Proposed improvements and alternative approaches to LLM suggestions. • Conducted comprehensive user research with 8+ pax and used multiple methods. • Created rich personas with emotional drivers, goals, and pain points • Used LLM to synthesize insights and validated all findings with real data. • Identified 5+ user problems with strong evidence. • Analyzed competitors thoroughly (5+ apps) and understand market opportunity. • Generated 6+ ideas using multiple ideation techniques with high creativity. • Created high-fidelity prototypes or interactive mockups if applicable. • Designed comprehensive user flows (happy path, edge cases, alternatives). • Conducted rigorous user testing with 8+ pax and made 3+ iterations with passing criteria. • Tested accessibility for diverse user types. • Comprehensive testing strategies with user testing. • Performance optimization with evidence of refinement. • Code is production-ready with professional quality.	• Implemented 5+ concepts successfully with production-quality code • Extensively evident of refactored code multiple times. • Created architectural solutions with clear scalability. • Conducted comprehensive user research with 8+ pax and uses multiple methods. • Created rich personas with emotional drivers, goals, and pain points. • Identified 5+ user problems with strong supporting evidence. • Analyzed competitors thoroughly (5+ apps) and understands market opportunity and positioning • Generated 6+ ideas using multiple brainstorming techniques with high creativity. • Evaluate and explain on ideas that would be targeted to address user needs. • Created high-fidelity prototypes or interactive mockups where possible. • Designed comprehensive user flows (happy path, edge cases, alternatives) • Conducted rigorous user testing with 8+ pax and made 3+ iterations with passing criteria. • Tested accessibility for diverse user types • Comprehensive testing strategy with user testing. • Performance optimization evident of refinement. • Code is production-ready with professional quality and clear documentation.

Component: Presentation (Individual 10%)

	Declared with LLM Assistance	**Declared without LLM Assistance**
Basic (0 ~ 1) “Know minimal”	• Very brief introduction and unclear of what app does. • Minimal feature description. • No explanation for why features were chosen. • Not able to explain code generated from LLM or how it works. • Not able to mention any MAD concepts. • App demo does not work smoothly or crashes. • Cannot answer most questions. • Shows minimal effort in presentation.	• Very brief introduction and problem statement unclear. • Minimal feature description. • No explanation for design choices. • Not able to explain code or features clearly. • Not able to mention any concepts or patterns. • App demo does not work or crashes. • Cannot answer most questions. • Appears unprepared.
Average (1 ~ 2) “State and explain”	• Clear introduction and explains main problem. • Describe features and what they do. • Gives basic reasons for feature choices. • Explain which parts were from LLM and make changes. • Can explain what features do. • Mentions 1-2 MAD concepts during presentation and explain the link to the app. • App demo works for main features ONLY. • Answers some questions with basic explanations • Shows understanding of what was built but limited on why it was built that way.	• Clear introduction and explains main problem. • Describe features and their purposes. • Give basic reasons for implementation choices. • Can explain code relationships to features. • Mentions 1-2 MAD concepts during presentation and explain the link to the app. • Explains basic code organization. • App demo works smoothly for main features ONLY. • Answer some questions confidently. • Shows understanding of what was built and why it was built that way.

Intermediate (2 ~ 3) “State, explain and analyse”	• Good introduction and explain problem with stakeholders. • Detailed description of features and why they are needed in the app. • Explain the reasons for feature choices. • Clearly explain which parts are generated by LLM and why changes were made to it. • Can explain LLM code improvements in detail • Mentions 2-3 MAD concepts and how it is used in the development • Discusses changes made to LLM code and improvements made with evidence. • App demo works smoothly and shows most features and edge cases. • Answer most questions well with good explanations. • Shows strong understanding of implementation and design choices.	• Good introduction and explain problem with stakeholders. • Detailed description of features and their value in the app. • Explain reasons for design and implementation choices made. • Mentions 2-3 MAD concepts and how it is used in code. • Explains code organization and basic architecture. • Discusses testing approach and fixes made if any. • App demo works smoothly and shows most features and edge cases. • Answer most questions well with good explanations. • Shows strong understanding of implementation and design.
Advanced (3 ~ 4) “State, explain, analyse and propose”	• Excellent introduction and clearly explain problems and user needs. • Detailed feature description with clear value for each feature developed. • Explains reasons for all feature choices with clear thinking and rationale. • Clearly explain LLM usage, significant customization, improvements and able to show refactoring of code. • Analyses LLM code/ suggestions and proposes better solutions. • Mentions 3+ MAD concepts and demonstrate how they work together. • Discusses major refactoring and quality improvements/ • App demo is smooth and professional which covers all features and edge cases well. • Answer all questions thoroughly with good technical depth.	• Excellent introduction and clearly explain problems and user needs. • Detailed feature description with clear value for each feature. • Explain reasons for all design and implementation choices. • Demonstrates 3+ MAD concepts and how they work together. • Clearly explain architecture and design patterns used. • Discusses testing strategy and performance considerations. • App demo is smooth and professional and covers all features and edge cases. • Answer all questions thoroughly with good technical depth.

	• Shows strong understanding of implementation, design, and architecture. • Propose improvements and alternative solutions other than those obtained from LLM.	• Shows strong understanding of implementation and architecture • Propose improvements and explain design decisions clearly.

Stage 2
Component: Implementation & Improvement (Individual 20%)

	Declared with LLM Assistance	**Declared without LLM Assistance**
Basic (0 ~ 1) “Stating as is”	• Implemented basic Activity without clear design • Used LLM code snippets directly without modification or understanding. • Feature depends heavily on implementations of other team members and cannot function independently. • Uses basic Stage 1 concepts with minimal extension and has no evidence of research or self-learning. • Copy-pasted LLM solutions without customization and/ or no new domain was implemented. • Minimal error handling, crashes on edge cases and evident that no testing or validation was conducted. • Poor code organization with/ without limited or no comments explaining logic. • Feature not validated with users and no consideration of user needs or pain points. • Cannot articulate why the feature is independent, necessary and/ or enhances the app. • No comprehensive documentation; missing feature descriptions, design rationale, or user guide	• Implemented basic Activity without clear design. • Minimal architectural consideration and tightly coupled code. • Feature developed depends on the implementations of other team members and cannot function independently. • Uses basic Stage 1 concepts with minimal extension and has no evidence of self-learning. • Minimal error handling and crashes on edge cases. • Code lacks comments or documentation and has unclear structure. • No new domain or advanced concepts implemented. • No prelim testing or validation conducted. • Feature not validated with users and has no consideration of user needs. • No comprehensive documentation and/ or missing descriptions and design rationale

Average (1 ~ 2) “State and explain”	• Implemented appropriate Activity/Service for the app. • Basic to moderate architectural structure and made minor customizations to LLM code generated. • New feature mostly stand-alone with minimal dependencies and can function with workarounds. • Extends Stage 1 concepts with basic new learning and shows some research and self-learning. • Identified and fixed 1-2 issues in LLM code and applied basic error handling for common scenarios. • Some commented code explaining logic and demonstrates basic programming practices. • Basic new domain or concepts implemented and has limited sophistication. • Code works with minor issues and basic documentation is provided. • Basic user research or feedback considered from 1-2 pax and feature addresses identified user needs. • Some informal testing or validation conducted with basic UX design. • Some evidence of design consideration but are basic and brief feature descriptions.	• Implemented appropriate Activity/Service for the app. • Basic to moderate architectural structure; appropriate design is evident. • Feature mostly stand-alone with minimal dependencies and can function with workarounds. • Extends Stage 1 concepts with basic new learning and shows some research. • Applied basic error handling for common scenarios and code mostly works. • Well-commented code and demonstrates basic good programming practices. • Basic new domain or concepts implemented. • Basic user research or feedback considered from 1-2 pax. • Some informal testing or validation and evidence of basic UX design. • Some evidence of design consideration and basic documentation

Intermediate (2 ~ 3) “State, explain and analyse”	• Implemented appropriate Activity/Service with good design suited for the app. • New domain implemented and demonstrated strong understanding of advanced concepts. • Good architectural structure with clear design patterns and significantly refactored LLM code • Feature completely independent with no dependencies on other implementations. • Extends Stage 1 concepts with strong new learning and research. • Fixed multiple issues in LLM code and refactor with comprehensive error handling for multiple scenarios. • Well-commented code with brief descriptions explaining major parts and design decisions. • Demonstrates good programming practices throughout and code is well-organized. • Strong user research or feedback considered with 4-6 pax and feature addresses real user needs. • Systematic testing and validation conducted with good and consistent UX design. • Made improvements based on feedback and demonstrate thoughtful UX considerations. • Good documentation of features, design, user guide and adds clear value to the app proposition.	• Implemented appropriate Activity/Service with good design suited for the app. • New domain implemented and demonstrates strong understanding of advanced concepts. • Good architectural structure with clear design patterns and makes appropriate design decisions. • Feature completely independent with no dependencies on other implementations. • Extends Stage 1 concepts with strong new learning and research. • Comprehensive error-handling for multiple scenarios and well-tested to certain extent. • Well-commented code with brief descriptions explaining major parts of the developed code. • Demonstrates good programming practices throughout and code is well-organized. • Strong user research or feedback considered with 4-6 pax and addresses real user needs. • Systematic testing and validation conducted with good and consistent UI/UX. • Made improvements based on feedback and displays thoughtful design considerations. • Good documentation of features, design and user guide.

Advanced (3 ~ 4) “State, explain, analyse and propose”	• Implemented complex and well-designed Activity/Service suited for the app. • New domain clearly implemented and somewhat demonstrates mastery of advanced concepts beyond Stage 1. • Sophisticated architectural structure with advanced design patterns and demonstrate extensively refactored LLM code. • Feature completely independent and zero dependencies on other implementations. • Extends Stage 1 concepts with sophisticated new learning and research. • Filled gaps where LLM knowledge is incomplete and created novel solutions beyond initial LLM output. • Comprehensive error handling for all scenarios and edge cases. • Well-commented code with clear descriptions for all major parts of code and explains design rationale. • Demonstrates excellent programming practices throughout and code is clean and organized. i.e. no code smells. • Performance optimized through robust app behavior • Comprehensive user research and feedback with 8+ pax and through multiple sources to addresses core user needs. • Rigorous testing and validation with well thought and consistent UX across all interactions throughout app. • Made multiple improvements based on data and demonstrate extensive design iteration • Comprehensive documentation of app features, design decisions and detailed user guide	• Implemented complex and well-designed Activity/Service suited for the app. • New domain clearly implemented and somewhat demonstrates mastery of advanced concepts. • Demonstrate architectural structure with advanced design patterns. • Feature completely independent and zero dependencies on other implementations. • Extends Stage 1 concepts with new learning and research. • Comprehensive error handling for all scenarios and edge cases. • Well-commented code with clear descriptions for all major parts which explains design rationale. • Demonstrates excellent programming practices throughout and code is clean and organized. i.e. no code smells. • Performance optimized with evidence of a robust app. • Comprehensive user research and feedback with 8+ pax through multiple methods and core user needs are addressed. • Rigorous testing and validation with well thought and consistent UI/UX across all interactions in app. • Made multiple improvements based on data and analysis and done extensive design iteration. • Comprehensive documentation of features, design decisions and detailed user guide. • User insights deeply integrated throughout with accessibility and diverse users considered

	• User insights deeply integrated for accessibility and diverse user needs considered.	

Component: Presentation (Individual 10%)

	Declared with LLM Assistance	**Declared without LLM Assistance**
Basic (0 ~ 1) "Stating as is"	• Very brief or unclear introduction to the app and feature. • Minimal description of app and its features and its purpose is unclear. • Little to no reasoning provided for implementation motivation. • Cannot illustrate how new concepts are applied in implementation. • Cannot explain LLM code generated, self-learning evidence and design decisions. • App demonstration does not work or has major issues or crashes frequently. • Cannot answer Q&A questions or responses are vague. • Demonstrates poor programming practices and overall code quality are of concern. • UX is inconsistent or confusing and/ or lacks thoughtful design. • Appear unprepared and have disorganized presentation. • Shows minimal understanding of own work or Stage 2 requirements.	• Very brief or unclear introduction to the app. • Minimal feature description and the purpose or the new implementation is not aligned with the app. • Little to no reasoning for implementation choices. • Cannot explain how new concepts are applied. • Cannot explain design decisions or self-learning. • Demonstrates poor programming practices. • UX is basic or inconsistent. • Code quality is of concern and lacks documentation. • App demo does not work or crashes frequently. • Cannot answer Q&A questions. • Appears unprepared and disorganized. • Shows minimal understanding.

Average (1 ~ 2) “State and explain”	• Clear introduction to the app and explains main feature and purpose. • Describes app and its features with clear explanations. • Provides basic reasoning for implementation motivation and explains why feature is chosen. • Mentions how new concepts learned are mentioned in implementation but not illustrated in depth • Can explain what LLM code does and basic design decisions. • Demonstrates some good programming practices • Basic UX that is functional but not particularly thoughtful. • Shows organized presentation with basic documentation. • Feature demo works for main use cases. • Answers some Q&A questions with basic technical explanations. • Shows understanding of what was built and basic Stage 2 requirements.	• Clear introduction to the app and explains main feature and purpose. • Describes app and its features with clear explanations, • Provides reasoning for implementation motivation. • Presents how concepts learned are implemented. • Can explain architectural structure and design choices. • Demonstrates good programming practices • Basic to functional UX that are understandable but simple • Well-commented code and basic documentation provided. • Presentation is somewhat organized. • Feature demo works for main use cases. • Answers Q&A questions with basic technical explanations. • Shows understanding of implementation and design.

Intermediate (2 ~ 3) “State, explain and analyse”	• Good introduction to the app and explains feature and context clearly. • Detailed description of app and its features with clear explanations are present. • Sound reasoning for implementation motivation and can explain design choices clearly. • Illustrates clearly how new concepts learned are applied in implementation. • Explain architectural structure and design pattern choices effectively. • Demonstrates good programming practices and code quality is evident. • Discusses good and consistent UX design with user feedback integration. • Demonstrate how concepts from MAD are implemented while showing understanding of application • Can explain design improvements and iterations made from LLM generated code. • Shows good presentation with technical depth and supporting evidence. • Feature demo works smoothly and has good and consistent UX demonstrated. • Answers most Q&A questions well with good technical explanations. • Shows strong understanding of implementation, design, validation, and/ or programming practices.	• Good introduction to the app and explains feature and context clearly. • Detailed description of app and its features with clear explanations. • Sound reasoning for implementation motivation and explains design choices. • Illustrates clearly how new concepts are applied in implementation. • Explains architectural structure and design choices effectively. • Demonstrates good programming practices and code quality is evident. • Discusses good and consistent UI/UX design. • Demonstrates how concepts from MAD are implemented and shows clear understanding. • Can explain design process and improvements made. • Shows good presentation with technical depth and evidence. • Feature demo works smoothly with good UI/UX. • Answers most Q&A questions well with good technical explanations. • Shows strong understanding of implementation, design and/or best practices.

Advanced (3 ~ 4) "State, explain, analyse and propose"	• Good introduction to the app with comprehensive explanation of feature context and strategic value. • Detailed description of app and its features with compelling explanations on value and context. • Sound reasoning for implementation motivation and well-justified design choices with evidence. • Excellently illustrates how new concepts learned are applied in implementation. • Explains sophisticated architectural structure with advanced pattern rationale • Clearly explains LLM strategy, extensive customization, gap-filling and novel solutions on top of LLM given code. • Analyzes and proposes improvements to LLM code and architectural approaches. • Demonstrates excellent programming practices and code quality and organization are evident. • Discusses well thought and consistent UX with user feedback deeply integrated. • Discusses how new advanced concepts are applied throughout implementation. • Discusses comprehensive user research with 8+ pax and how it validated design. • Discusses extensive iteration cycles with quantitative improvements and passing criteria. • Comprehensive documentation quality and detailed user guide provided. • Shows presentation with strategic depth. • Feature demo is smooth with robust behavior and excellent UI/UX. • Answer all Q&A questions thoroughly with deep technical explanations.	• Good introduction to the app and comprehensive explanation of feature context and strategic value. • Detailed description of app and its features with compelling explanations. • Sound reasoning for implementation motivation and well-justified choices with evidence. • Excellently illustrates how new concepts are applied in implementation. • Explains architectural structure with advanced pattern rationale. • Demonstrates excellent programming practices and code quality and organization are evident. • Discusses well thought and consistent UX design with proper user feedback integration • Discusses how new advanced concepts are applied throughout the app feature • Discusses comprehensive user research with 8+ pax across multiple methods employed and considered with design validation. • Discusses and demonstrates extensive iteration cycles with quantitative improvements. • Comprehensive documentation with quality and detailed user guide provided. • Presentation with strategic depth and explanations. • Feature demo is smooth with robust behavior and excellent UI/UX. • Answer all Q&A questions thoroughly with deep technical explanations.

	• Provides alternative approaches and demonstrates critical thinking. • Shows somewhat level of implementation, design, validation and/or programming practices. • Demonstrate systems and architectural design while aware of market. i.e. Market awareness	• Provides alternative approaches and demonstrates critical thinking in implementation and deployment of the feature. • Shows somewhat level of implementation, design, validation and/or programming practices • Demonstrates systems and architectural maturity while keeping it with user-centered development.