Midterm Exam – Key Instructions

1. **Mode & time:** Take-home exam to be solved **INDIVIDUALLY** choose **ONE** of the six given problem options. You have about **36 hours**; **deadline is 11:59 pm, 7th December (PKT)**.
2. **Submission (primary):** Submit a **Git repository link** (to be shared via MS Teams). All required files and code must be in this repo.
3. **Late penalty:** After 11:59 pm, **10 marks deducted per 6 hours (or part thereof)** (even 1 minute late = 10 marks deduction).

Physical copy

4. Use the **exam copy** for sketches (especially UI/futuristic UI) and notes.
5. Return this paper in the **Drop Box at the President Office** on **Monday before 10:00 a.m.** This is **compulsory**, in addition to the Git submission.

AI tools & integrity

6. **AI tools are allowed**, but use must be **fully documented** in AI-log.md:
    - List all tools used.
    - Include all prompts.
    - Mention what each prompt was used for.
7. You may take **suggestive help**, but the work must be **understood and owned by you**.
8. **Strictly forbidden (straight F):**
    - Copying code/content from classmates or others without attribution.
    - Sharing your solution.
    - Submitting anything you **cannot explain** in a viva.

Required files in Git repo

For the chosen problem option, include at least:

9. ProblemStatement.md – user, pain point, why it matters, MVP goal, scope and out-of-scope, assumptions.
10. UseCases.md + short design – at least **3 use cases** (actor, trigger, preconditions, main flow, alternate flow) and a simple high-level design/data-flow.
11. Code + README.md – working MVP, how to run it, example input/output.
12. TestPlan.md – 3–5 test cases (normal, positive, negative, edge), with input, expected result, actual result.
13. AI-log.md – tools used, prompts, what they were used for, brief reflection on help vs risks.
14. UI-Sketch-and-Vision (file or .md) – UI sketch/wireframe for a future, more complete product, plus a short explanation of main UI ideas.
15. ReleaseRoadmap.md – brief plan for **3 months**, **1 year**, **2 years**: how the product could evolve (features, integrations, improvements).

How you will be judged

16. Marks focus on **effort, clarity, and understanding** of practical software engineering:
    - Problem framing and scope.
    - Use cases and design.
    - Working MVP and basic testing.

- Honest, reflective AI use.
- Product thinking (future UI and release roadmap).markdownguide+2

17. It is better to submit **partial but genuine work you can explain** than a "perfect" solution you do not understand.

There are six real-life problem options, each with a clear paragraph description. You may choose any ONE for the 100-mark midterm.

Six problem statements (Choose Any One)

Option 1 – HR CV–JD Match Assistant
Design an HR assistant that helps a recruiter quickly see how well a candidate's CV matches a specific advertised job. The system should accept a CV and a Job Description (JD), extract key elements such as skills, experience, and qualifications, and compute a simple match indication (for example, a percentage score or Low/Medium/High). It should also highlight the main reasons for this assessment, such as strong overlaps or missing key skills. Design a minimal tool that can support a recruiter's initial screening and help candidates understand how well they fit in a role.

Option 2 – Learning Path Recommender
Design a learning assistant that helps a user move from their current skill level to a defined learning goal in a specific domain (for example, "become a junior web developer" or "learn the basics of data analysis"). The system should take as input the user's current skills, background, and time availability, and then output a short, prioritized learning path and a simple 4–6 week roadmap. The roadmap may contain pointers to specific contents (videos, sites, texts, etc). The goal is to provide a focused, realistic sequence of topics or activities rather than a huge list of resources, so that the user can see a clear starting point and immediate next steps.

Option 3 – Study Session Planner Assistant
Design an assistant that converts a short-term academic goal into a concrete study schedule over the next 1–2 weeks. The user specifies what they need to prepare for (such as an exam or project milestone), which topics are involved, and what time slots they have available each day. The system should then generate a simple plan that assigns topics to specific days and sessions, balancing the workload and ensuring at least some revision time for more difficult topics. The focus is on turning vague intentions into an actionable schedule a student can realistically follow. Keep University students in mind of any discipline.

Option 4 – Real-Estate Match Advisor
Design an assistant that helps a home seeker or real-estate agent understand how well available properties match a buyer's preferences. The user specifies their budget range, preferred areas, required features (such as number of bedrooms, parking, or proximity to workplace), and a small list of candidate properties (e.g. from Zameen.com or Zillow.com). The system should compute and display a match level

for each property and briefly explain which criteria are satisfied and where compromises are needed. The aim is to support transparent, quick comparison of options rather than to replace professional advice. The advisory could be for both rental or purchase.

Option 5 – Best-Fit Car Finder
Design an assistant that recommends suitable car options for a user, based on budget and usage needs. The user provides constraints such as price range, new or used preference, fuel type, and main purpose (for example, daily city commuting, family trips, etc). The system works with a small catalog of cars (or from sources like pakwheels.com), calculates a simple ranking or score for each option, and presents the top suggestions with short justifications (such as "fits budget and fuel preference but has limited trunk space"). The aim is to make trade-offs visible and help users quickly narrow down to a few choices.

Option 6 – Personal Time & Habit Coach
Design a "habit and time-use" assistant that helps a user reflect on how they spend their day and suggests small, positive adjustments. The user logs their activities for a day (or a typical day) with categories and durations, such as work, social media, study, exercise, family time, and sleep, along with one or more personal goals (for example, "more mindfulness", "more reading", "less social media"). The system summarizes how time is distributed, surfaces simple insights (for example, areas of over-investment or neglect), and proposes a few micro-changes like shifting some minutes towards productive or mindful activities. The focus is on awareness and realistic, incremental improvement rather than strict habit enforcement.

Deliverables required

You must include the following in their Git repository for the chosen option:

A. Markdown file for Problem Statement, **ProblemStatement.md**.
   To know more about Markdown files, refer here (or any other source you like)
   https://www.markdownguide.org/getting-started/
   The problem statement must address:
   a. Pain point, primary users, and why the problem matters.
   b. Scope and explicit out-of-scope features.

B. **Use Cases & Design**
   - At least 3 detailed use cases.
   - Simple high-level design / data-flow description.

C. **MVP Implementation (code)**
   - Working minimal system for the chosen problem.
   - README.md with run instructions and example input/output.

D. **TestPlan.md**
   - 3–5 test cases with input, steps, expected and actual outcomes.
   -

E. **AI-log.md**
   - All AI tools used, prompts, and brief reflection on usefulness and limitations.

F. **UI Sketch & Vision**
   - A simple UI sketch or wireframe (image or markdown-based ASCII/PlantUML) showing how the "final product" UI might look if this system were fully developed beyond the MVP.
   - 1–2 paragraphs explaining key UI decisions (for example, how information is organized, how the user flow supports the pain point).

G. **Release & Evolution Plan**
   - A short "release roadmap" (for example, 3-month, 1-year, 2-year view) describing how the product could evolve over time.
   - At each stage, a few bullet points of planned improvements (for example, better algorithms, more data sources, richer UI, integrations, analytics), to emphasize product longevity and lifecycle thinking.

Marks breakdown (100 marks)

Component	Marks
Problem & pain point (ProblemStatement.md)	15
Use cases & basic design	15
MVP implementation (working code + README)	30
Testing plan & evidence	5
AI tools usage & reflection (AI-log.md)	10
UI sketch & future UI vision	10
Release & evolution roadmap (3m / 1y / 2y)	15
Total	100

- **Passing mark:** 40 and above
- **Indicative Grades:**
  - Band of A's: 80 and above
  - Band of B's: 70-79.9
  - Band of C's: 55-69.9
  - Band of D's: 40-54.9
  - Below 40's: FAIL

Wishing you all the Best