ChatGPT

IAU Life Simulator – Requirements & Timeline

Project Overview

The **IAU Life Simulator** is a pixel-art university-life management game built in **Lua** using the **Love2D** framework.
Players take on the role of a college student who must balance studying, part-time work, social activities and health to progress through semesters.
Instead of combat, the game uses exams and projects as “battles” where the student’s preparation and stats (knowledge, energy, mood, health and money) determine success.
This project aims to provide a fun and educational experience that mirrors the challenges of managing real student life.

Team

Team Member	Role
Ammar Bunajmah	Team leader, backend & frontend developer
Abdulsamad Alhassan	Frontend developer
Salem Balkhair	Art design
Omar Bodai	Backend developer
Dhiyaa Madkhali	Programming debugger & backend developer
Mohammed Al Shehab	QA tester

Requirements & Scope

Included Features

The following features are considered **within scope** for the first prototype:

- **Core simulation loop**: manage stats such as knowledge, energy, social life, fitness, stress and money.
- **Daily/weekly planning system**: allocate time blocks (morning/afternoon/evening/night) to actions like study, work, gym, rest and socializing.
- **Exam mechanic**: exams and projects act as the main progression checks, using stats to determine outcomes.
- **User interface**: basic menus and HUD to display available actions, current stats and results.
- **Save/load system**: allow players to resume progress.

Excluded Features

The prototype will **not** include the following:

- Multiplayer or online features.
- Advanced graphics or animations beyond simple 2D sprites.
- Large storylines or open-world exploration.

Project Goals

The goals of the project are to:

- Develop a **fun and educational** management RPG that reflects student life.
- Let players experience the challenge of balancing study, work and social life.
- Implement a **turn-based exam system** inspired by passive-combat games.
- Practice software engineering principles: planning, design, modular coding and testing.
- Deliver a polished prototype that can be expanded with items, random events and NPCs.

Main Development Tasks

According to the project idea form, the major development tasks include:

1. **Game Design** – define the player stats, available activities, exam mechanics and progression rules.
2. **Programming** – implement the core simulation loop, user interface, exam logic and random events.
3. **Art & Assets** – create pixel-art sprites and icons for stats, activities and interface.
4. **Testing** – debug gameplay balance (exam difficulty, activity effects) and fix bugs.
5. **Documentation** – produce design documents, technical diagrams and a user manual.

Tools & Technologies

- **Language & engine:** Lua with the Love2D framework (lightweight, cross-platform).
- **Graphics:** simple 2D pixel art using sprite editors (e.g., Aseprite).
- **Version control:** Git/GitHub for source management and task tracking.
- **Development platforms:** cross-platform (Windows, macOS, Linux).
- **Libraries:** optional Love2D UI frameworks to simplify menu and HUD development.

Development Timeline & Milestones

The timeline below is a suggested schedule for delivering a polished prototype. Each phase lists major activities and an estimated duration. All dates are in **2025** and use the Asia/Riyadh time zone; adjust as needed based on your academic calendar and workload.

Phase	Activities (keywords)	Dates	Duration
Planning & Design	define stats, activities & exam system; design UI wireframes; assign tasks	**Oct 14 – Oct 27**	2 weeks

Phase	Activities (keywords)	Dates	Duration
Prototype Development	implement core simulation loop, daily/ weekly planner & basic exam mechanic	**Oct 28 – Nov 10**	2 weeks
Art & UI Implementation	create sprites/backgrounds/UI elements; integrate art into prototype	**Nov 11 – Nov 24**	2 weeks
Feature Completion	add random events, save/load, refine exam system & balance gameplay	**Nov 25 – Dec 8**	2 weeks
Testing & Polishing	QA testing, bug fixes, difficulty balancing & performance improvements	**Dec 9 – Dec 20**	2 weeks
Documentation & Final Review	prepare final documentation, user manual & presentation; finalize prototype	**Dec 21 – Dec 28**	1 week

Note: These time estimates assume a part-time student project with weekly meetings. Teams can adjust durations depending on academic deadlines and resource availability. The phases are inspired by typical software project templates.

Next Steps

1. **Finalize design documentation:** Draft a detailed game design document covering stat mechanics, actions, UI wireframes and exam logic. This will guide development and ensure everyone agrees on the scope.
2. **Set up repository and task board:** Initialize the GitHub repository, choose a branching strategy and create a project board (e.g., GitHub Projects or Trello) to track tasks and milestones.
3. **Begin prototype implementation:** Start coding the core simulation loop and exam system while art assets are drafted in parallel.
4. **Weekly reviews:** Hold weekly stand-ups to monitor progress, identify blockers and adjust the schedule. Use the status report template to document progress and issues each reporting period.
5. **Prepare for playtesting:** Once the core loop is functional, compile a test build and gather feedback from classmates or potential players to refine balancing and user experience.

This document synthesizes information from the Project Idea Form and general project planning templates. It outlines clear requirements, goals, scope and a realistic timeline to guide development. Feel free to revise dates and tasks as the team gains more clarity on workload and academic commitments.