Kontroller

Software Development Plan

Version 3.0

Revision History

| **Date** | **Version** | **Description** | **Author** |
| --- | --- | --- | --- |
| 02/06/2025 | 1 | Drafting the first version of the report | Nguyen Gia Nghi |
| 03/06/2025 | 2 | Reviewing and adjusting | All members |
| 23/07/2025 | 3 | Revise all Project Plan | All members |

Table of Contents

[**1.**](#_heading=h.x6j34oox2ldl) **Introduction 4**

[**2.**](#_heading=h.ouy6rbm8f1lm) **Project Overview 4**

[*2.1*](#_heading=h.mcz5tek3u5tz) *Project Purpose, Scope, and Objectives 4*

[*2.2*](#_heading=h.94e55y5osvn5) *Assumptions and Constraints 4*

[*2.3*](#_heading=h.3rglwg7v42ic) *Project Deliverables 4*

[**3.**](#_heading=h.3h5i7yca6gyd) **Project Organization 4**

[*3.1*](#_heading=h.tpacl943ds56) *Organizational Structure 4*

[*3.2*](#_heading=h.x4p3tfrnksiz) *Roles and Responsibilities 4*

[**4.**](#_heading=h.nms3eh2kf96g) **Management Process 4**

[*4.1*](#_heading=h.hhxu7wpv5sae) *Project Estimates 4*

[*4.2*](#_heading=h.4w8uywd3bhap) *Project Plan 4*

[4.2.1](#_heading=h.lybp1cazmh2m) Phase and Iteration Plan 5

[4.2.2](#_heading=h.mnjzjbwyuacm) Releases 5

[4.2.3](#_heading=h.bbx7dbcii0pm) Project Schedule 5

[4.2.4](#_heading=h.9ag8xozdgxt2) Project Resourcing 5

[*4.3*](#_heading=h.69z09abume25) *Project Monitoring and Control 5*

[4.3.1](#_heading=h.qr0nbhh9ewnq) Reporting 5

[4.3.2](#_heading=h.z9epa61ne1sf) Risk Management 5

[4.3.3](#_heading=h.41dzim8121bn) Configuration Management 6

Software Development Plan

# Introduction

Kontroller is a cross-platform web application that allows users to log, rate, and review video games while sharing

their gaming experiences with friends and people in the gaming world. The goal is to provide a centralized platform similar to Letterboxd or RateYourMusic, but for games and gamers, leveraging real-time data, social features, and user-generated content. The initial product is a web-based MVP that supports both macOS, Windows and other platforms.

# Project Overview

## Project Purpose, Scope, and Objectives

**Purpose:**

* To create a user-friendly platform for gamers to track, share, and reflect on their gaming experiences, while fostering social interaction and community engagement.

**Scope:**

* Platform: Web-based, mobile-responsive MVP.
* Features: Game logging, reviews, custom lists, social networking, timelines, multiplayer tagging, and annual/monthly recaps.
* User Base: Casual to hardcore gamers, streamers, and content creators.
* Future: Designed to allow future expansion, such as gamification, recommendation engines, and third-party integrations (Steam, PSN, ect.).

**Objectives:**

* Allow users to log and review games with detailed notes and scores.
* Provide personalized features such as timelines, recap summaries, and custom lists.
* Enable multiplayer tagging and social sharing to enrich user interaction.
* Design a clean, responsive UI that works seamlessly across desktop and mobile.
* Build a foundation for future features like gamification, recommendation engine, or third-party API integration (e.g., Steam, PSN).

## Assumptions and Constraints

**Assumptions:**

* The target users have stable internet access and use modern browsers (e.g., Chrome, Firefox, Safari).
* Users are familiar with basic gaming platforms (Steam, PlayStation, mobile, etc.).
* Team members have intermediate to advanced skills in front-end and back-end development.
* Core features will be prioritized based on user interviews and MVP (Minimum Viable Product) scope.
* Feedback will be gathered iteratively during development (e.g., via user testing or internal demo).
* Open-source libraries and public APIs can be used where needed.

**Constraints:**

* Schedule constraint: Fixed timeline of 12 weeks for full delivery.
* Budget constraint: Zero-budget project — no external funding, all tools must be free or open-source.
* Staffing constraint: Team includes 5 members with defined roles (e.g., 2 developers, 1 designer, 1 tester, 1 project lead); no new members will be added.
* Scope constraint: Only core functionalities (logging, reviewing, timeline, social features) will be implemented in this phase.
* Technical constraint: Must be deployed on free-tier platforms (e.g., Vercel, Firebase, Supabase).
* Device constraint: Optimized primarily for Chrome and mobile web; native app not included in current scope.

## Project Deliverables

| **Artifact** | **Description** | **Target Delivery Date** |
| --- | --- | --- |
| **Project Proposal** | Overview of project purpose, scope, assumptions, and goals. | Week 1 |
| **Wireframes & UI Mockups** | Low-fidelity sketches and high-fidelity designs for core pages (home, game log, profile, etc.). | Week 2 |
| **System Architecture Diagram** | Diagram showing app structure, tech stack, and data flow. | Week 3 |
| **Functional Specification Document** | Detailed list of features and their behavior (login, log game, review system, timeline, etc.). | Week 3 |
| **Front-end Prototype** | Interactive prototype (React/Next.js or equivalent) with static data. | Week 5 |
| **Back-end Setup** | Basic database schema, API routes, authentication setup. | Week 6 |
| **MVP Web Application** | Fully functional app with core features (log, review, timeline, user accounts). | Week 9 |
| **Testing Report** | Bug log, usability testing results, user feedback summary. | Week 10 |
| **User Manual / Guide** | Simple documentation or onboarding guide for end users. | Week 11 |
| **Final Presentation / Demo Video** | Walkthrough of app features and project outcomes. | Week 12 |
| **Project Retrospective Report** | Summary of what went well, lessons learned, and future improvements. | Week 12 |

# Project Organization

## Organizational Structure



## Roles and Responsibilities

• Frontend Developers: Build user interfaces and integrate IGDB game data for display and

interaction.

•Backend Developers: Design the system and software architecture, including UML diagrams and API documentation. Develop features for user authentication and APIs for communication between the frontend and backend. Support frontend developers by designing clear, maintainable APIs and implementing backend logic that interacts efficiently with the server and database. Ensure that data querying is accurate, scalable, and follows clean code principles to facilitate future maintenance and expansion. Deploy the backend infrastructure with extended capabilities for server management, and integrate both the database and frontend with the backend system.

• Database Engineer: Design, implement, and manage the app’s database system. Work with backend developers to ensure data integrity.

• Project Manager: Oversee sprint planning and task distribution, ensuring the tasks are distributed evenly among team members. Manage the issues while completing each particular task

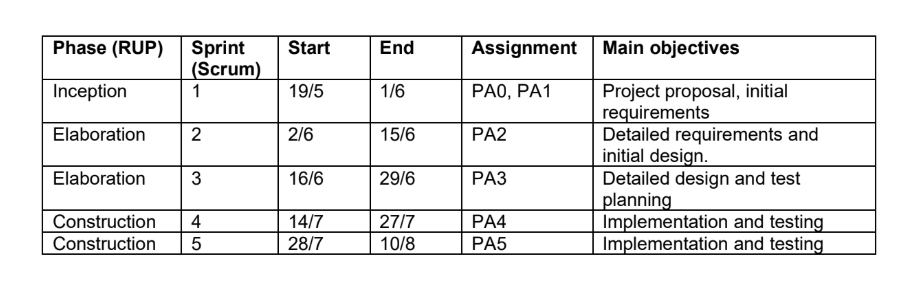
| ***Person*** | ***Role*** |
| --- | --- |
| *Nguyễn Gia Nghi*  *Nguyễn Minh Thuận & Dương Đức Thịnh*  *Hoàng Ngọc Tùng*  *Nguyễn Thành Đạt* | * *Team lead (project manager)* * *Backend developer* * *Frontend developer* * *Database developer* |

# Management Process

## Project Estimates

## Project Plan

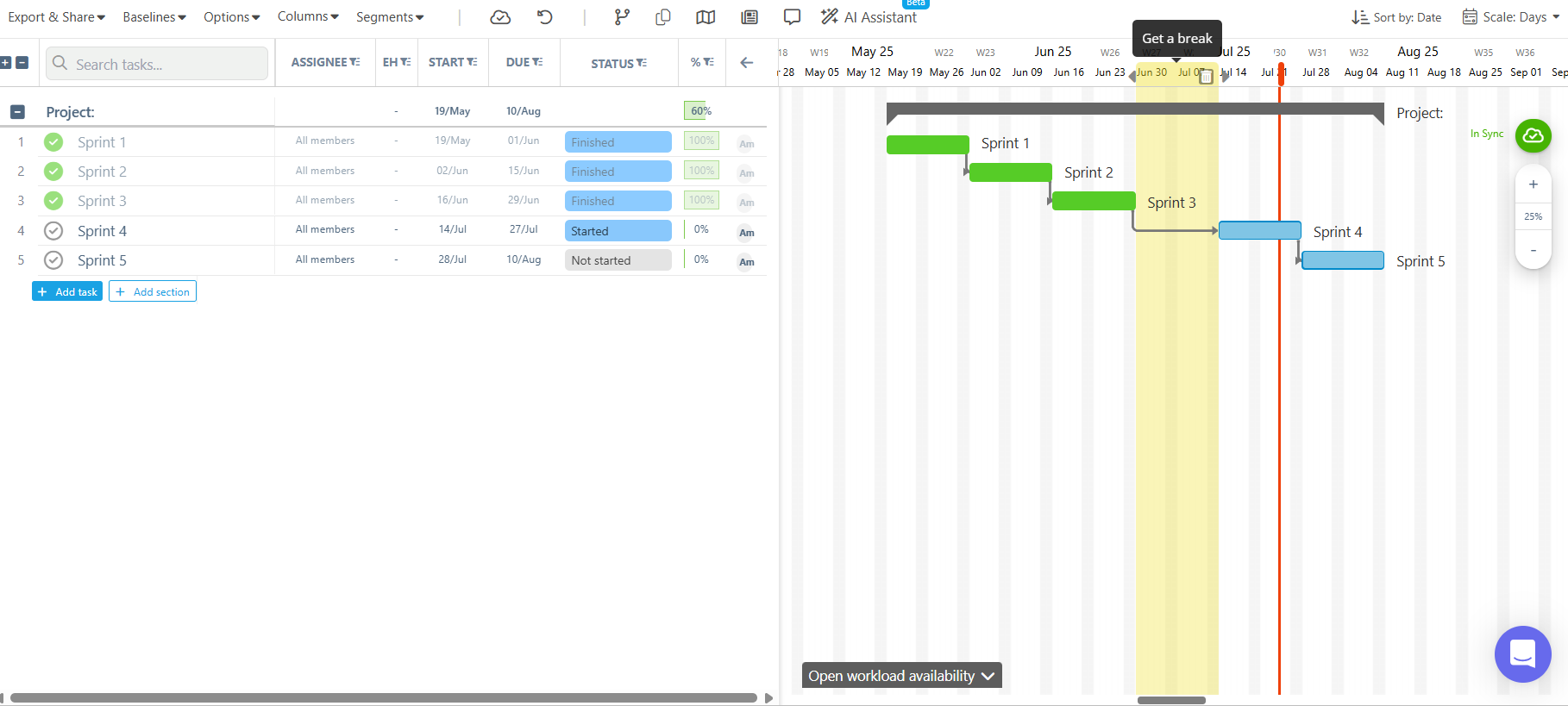
### Phase and Iteration Plan

**

### Releases

* **Release 1 (Sprint 2)**: Internal testing of basic system architecture and UI/UX prototype (alpha)
* **Release 2 (Sprint 3)**: Demo for design validation and test planning
* **Release 3 (Sprint 4)**: First build with core functionalities (beta)
* **Release 4 (Sprint 5)**: Final release with completed features (stable demo)

### Project Schedule



## Project Monitoring and Control

### Reporting

* Weekly Meetings: Scheduled every Sunday evening via Google Meet for sprint planning, task updates, and progress review.
* Weekly Status Reports: Each member is responsible for completing their assigned tasks and reporting on their progress. The team then collaboratively compiles all individual updates into a comprehensive team report, which is reviewed and finalized together before being shared on Slack and stored on Google Drive.
* Informal Chats: daily coordination and problem-solving via Slack and Messenger

### Risk Management

| *Risk ID* | *Risk Description* | *Probability* | *Impact* | *Risk Exposure* | *Priority* | *Mitigation Strategy or Contingency Plan* |
| --- | --- | --- | --- | --- | --- | --- |
| R1 | Cannot complete assigned tasks and the product on time | Medium | High | High | High | Reviewing every week the achievements and determine exactly the goals so that tasks could be completed on time, project manager will closely observe and ensure the working process |
| R2 | Data from IGDB can not be retrieved or is limited | Medium | High | High | High | Having backup data, implementing APIs to retrieve data from IGDB if possible |
| R3 | Lack of coordination between data engineer and backend developers | Medium | High | High | High | Data engineer and backend developers will work closely with one another to ensure the validity of codes and data |
| R4 | Forget some crucial windows and features when handling frontend | Low | High | Medium | Medium | Determine and plan the design ahead to not leave out any vital departments of the app, carried out by PA and frontend developer |
| R5 | Github merging branch problems and data loss | Low | Medium | Medium | Medium | Reviewing Git commits regularly, using branches to preserve codes if necessary. |

### Configuration Management

* Source code and version control:
  + Git via GitHub for managing and tracking source code and commits.
* Document sharing and collaboration:
  + GitHub and Google Drive for storing documentation, planning materials, and presentation files.
  + Jira for weekly scheduling.
  + Slack and messenger for communication.
  + Figma for designing