Project: Web-based Manga Platform with OCR and Translation
Course: Introduction to Software Engineering (HCMUS) – Instructor: Mr. Truong Phuoc Loc
Meeting Date: November 3, 2025
Meeting Phase: Requirements Analysis Planning Meeting

Meeting Details

- **Time:** 14:00 – 16:00
- **Location:** Online (Discord meeting)
- **Attendees:** All group members (Group 07)
  - Le Anh Duy – 23127011
  - Tran Gia Huy – 23127199
  - Nguyen Thanh Luan – 23127296
  - Nguyen Thanh Tien – 23127539

Purpose: Formally kick off the Requirements Analysis phase, clarify project objectives and scope, gather and discuss all functional and non-functional requirements, and assign actions for producing the Software Requirements Specification (SRS) and related deliverables for the project.

Agenda

1. **Project Overview & Objectives** – Revisit the project vision, goals, and context.
2. **Stakeholders & User Roles** – Identify end-users and roles (Reader, Uploader, Admin, etc.) and their needs.
3. **Functional Requirements Gathering** – Brainstorm and list all features and behaviors the system must support.
4. **Non-Functional Requirements (NFRs)** – Identify quality attributes (performance, security, usability, etc.) and constraints.
5. **Use Case Identification** – Outline key use cases/user stories covering interactions between users and the system.
6. **Prototype Planning** – Discuss creating simple UI prototypes for critical use cases to validate requirements.
7. **Deliverables & Timeline** – Define outputs of this phase (requirements documents, prototypes) and deadlines.

8. **Action Items Assignment** – Assign responsibilities for drafting the SRS sections (functional list, use cases, etc.) and set the next meeting date.

Discussion and Notes

1. Project Overview & Objectives

The meeting opened with a reiteration of the project's vision and objectives. The team confirmed that the project is a **web-based manga reading platform with integrated optical character recognition (OCR) and machine translation**, aimed at eliminating language barriers in reading manga. We reviewed that the system will consist of two primary subsystems:

- a **Reader subsystem** for end-users to browse and read manga pages with an automatic per-page translation option, and
- an **Admin (Uploader) subsystem** for managing content uploads, approvals, and overall platform moderation.

It was emphasized that the platform should provide a **smooth, interactive reading experience** while handling dynamic content (manga pages, comments) and ensuring secure user access and data storage. The team noted the importance of real-time text recognition and translation for each manga page as a core feature of the platform's appeal. Additionally, the project's context in the *Introduction to Software Engineering* course means we must follow software engineering best practices and produce all required documents/artifacts for each phase.

2. Stakeholders & User Roles

We identified the key user roles and stakeholders of the system:

- **Readers:** General users who browse the website to read manga. They need features like account registration, viewing manga chapters with translations, saving favorites, tracking reading progress, commenting, rating, and sharing content.
- **Uploaders:** Users who provide content by uploading manga chapters. In this project, uploaders are essentially a role that readers can apply for and obtain upon admin approval. Uploaders require functionalities to submit new chapters with metadata, edit or manage their uploads, and view the status of their submissions.
- **Admins:** Administrators who oversee the platform. They manage content and users – approving or rejecting uploaded chapters, highlighting featured manga, managing user roles (e.g., granting uploader permissions), and possibly suspending accounts

for security. Admins also handle platform settings like featured content duration and ensuring content rules are followed.

- **Directors/Management:** The team discussed an implicit stakeholder (referred to as "director" in some requirements) representing project or business management who might need summary information like site analytics or revenue reports. While not an interactive end-user of the platform, this role influences requirements for administrative reporting features (e.g., viewing user visit statistics or revenue). We noted that such features might be implemented as part of the admin interface or a separate management dashboard.

Understanding these roles helped ensure our requirements cover all perspectives. The team agreed that Readers, Uploaders, and Admins are the primary user types we will focus on, with "director" needs being handled via admin reporting features if needed.

3. Functional Requirements Gathering

Next, the team brainstormed and listed all functional requirements (FRs) — the specific features and capabilities the system must provide. We reviewed each major function in terms of user stories (from the project proposal and our discussion) and ensured they align with the project goals. The key functional requirements identified include:

- **User Account Management:** Users can register and log in (including third-party OAuth options like Google/Facebook). The system must securely handle authentication and ensure user credentials are protected (passwords encrypted, data via HTTPS). An admin can assign roles to users (e.g., promote a user to Uploader role), and users should be able to request an uploader role upgrade.

- **Manga Browsing & Reading:** Readers can browse available manga titles, search by various criteria (title, author, genre, year) and see results quickly. We noted the requirement that search results should return in under 2 seconds for a good user experience. For reading, the system will display manga pages sequentially with options to navigate between chapters and pages.

- **Automatic Page Translation:** A core feature discussed is the ability for readers to get on-the-fly translations of manga text via OCR and machine translation. When a reader views a page, the system will offer an "Translate" option that sends the page image to the OCR/translation service and overlays the translated text for the user. The team agreed to integrate a third-party OCR/translation API for this (as building a custom OCR/MT solution is out of scope). We set a performance expectation that each page's translation should process in a reasonable time (around **≤5 seconds per page** as initially proposed).

- **Content Upload & Management:** Uploaders (and Admins) can upload new manga chapters. This involves providing required metadata (title, genre, summary, etc.) and the chapter's image files. Uploaded chapters will enter a pending state requiring Admin approval. Admin users will have an interface to review pending uploads, inspect content, then approve or reject each chapter. Once approved, the chapter becomes publicly readable. The admin interface will also allow admins to edit or delete manga entries, manage metadata, and oversee all content.
- **Content Moderation & User Management:** Admins can lock or suspend user accounts to enforce platform rules or security. They can also feature or highlight certain manga on the homepage (for promotions or popular content) for a limited time period. The meeting noted to include the ability for Admin to set "highlight" tags on content with an expiration (default 7 days, configurable).
- **Payment and Access Control:** The platform will include a payment mechanism for paid content. Readers should be able to pay a fee (via supported methods like credit card or local e-wallets like ZaloPay) to unlock certain manga chapters. After payment, access is granted to the content. The system must process transactions securely and relatively quickly (target under 5 seconds per transaction). We acknowledged that implementing a full payment gateway is complex; as a student project, integration might be simulated or limited to demonstrating the flow.
- **Community Features:** To enhance user engagement, the platform will allow readers to **comment** on chapters and **rate** them. Comments should appear immediately upon submission and the system should filter out offensive content. Users can also have personal **favorites lists** to save manga they like, and the system will remember their favorite titles and reading progress (so they can resume where they left off). Additionally, a **social sharing** feature will enable users to share links to manga on social networks, with one-click sharing options (Facebook, Zalo, etc.).
- **Analytics & Reporting:** Aligning with the "director" stakeholder needs, the system should gather basic analytics such as the number of users or site visits, and revenue reporting. An example discussed was a dashboard for admins/directors to view monthly and yearly revenue, and user traffic statistics. We agreed this is a lower priority compared to core features but should be kept in mind (perhaps implemented as a simple stats page for admins showing aggregated data, if time permits).

Throughout this discussion, team members cross-referenced the preliminary list of features from our project proposal to ensure nothing was missed. By the end, we compiled a comprehensive list of functional requirements covering account management, content management, reading & translation, community interactions, and payment.

4. Non-Functional Requirements (NFRs)

The team then identified and documented the non-functional requirements – the quality attributes and system constraints that the platform must satisfy. Key NFRs agreed upon include:

- **Performance:** The website should be responsive and efficient. Specific targets were recalled from the proposal:
    - Page load and basic operations should be fast (e.g., search results should return within 2 seconds).
    - Translation processing should ideally be ≤5 seconds per page to not frustrate users.
    - Administrative reports (if implemented) should generate in under 5 seconds as well for quick viewing of statistics.
    - The site should handle multiple concurrent users without significant degradation in response time. We acknowledged performance will also depend on the third-party APIs (OCR/translation), so we must handle those calls asynchronously to not block the UI.
- **Security:** Security is critical since user accounts and payments are involved. The system must enforce strong security practices:
    - All sensitive data transmissions (login, payments) use HTTPS encryption.
    - Passwords must be stored securely (hashed with one-way encryption, e.g., bcrypt).
    - User roles and permissions should be enforced on the backend to prevent unauthorized actions (only admins can approve content or assign roles, etc. – a point also noted as a functional requirement).
    - We also discussed protecting the platform from common web vulnerabilities (SQL injection, XSS, etc.) and ensuring the OCR/translation API keys or credentials are kept secure.

- **Usability:** The platform should be easy to use and navigate. We aim for a clean, intuitive UI design with consistent layout for reading and admin interfaces. It must be mobile-responsive, given many users may access via smartphones. (We recalled that our chosen frontend framework Next.js with Tailwind CSS will help ensure a responsive design across various devices.)
- **Reliability & Availability:** The system should be stable – crashes or downtime should be minimized. In practice, as a student project, 24/7 uptime is not a strict requirement, but we will design with error handling so that the application fails gracefully if the OCR service or payment service is unavailable. Data integrity is important: user data (comments, progress, transactions) should not be lost or corrupted.
- **Scalability & Maintainability:** While our project scale is small, we envision the design to be scalable to more users and content. The architecture should separate concerns (e.g., a three-tier architecture) to allow independent scaling of the frontend, backend, and database if needed. Code should be modular and well-documented for easy maintenance. The choice of technologies (React/Next.js, Node.js, PostgreSQL) was also made for their scalability and community support.
- **Compatibility:** The web app should work across modern browsers (Chrome, Firefox, Edge, Safari) and not be tied to any proprietary technology. It should also accommodate different screen sizes (desktop and mobile) smoothly, which ties into our use of responsive design.

These NFRs were recorded to be included in the SRS. The team agreed that meeting the functional requirements is the priority, but we will keep these quality goals in mind throughout design and implementation. In particular, performance and security will guide some technical decisions (e.g., using efficient database queries, caching where appropriate, and using established authentication libraries).

5. Use Case Identification

Using the functional requirements as a basis, we outlined the primary **use cases** for the system, essentially mapping how each user role interacts with the system to accomplish specific goals. The major use cases discussed include:

- **Register an Account / Login:** A user creates a new account or logs in to an existing account to become an authenticated Reader. (Precondition: user provides valid email/password or uses OAuth; Postcondition: user session begins, and they can access reader features.)

- **Browse and Search Manga:** The Reader searches or filters manga by criteria (title, genre, etc.) and browses the list of available titles. This encompasses the search performance requirement and listing results.
- **Read a Manga Chapter:** The Reader opens a manga chapter to read. The system loads the chapter's pages sequentially. Here we also incorporate sub-flows for "Translate page to language X" – the user triggers the OCR/translation on a page and sees the translated text overlay.
- **Manage Reading List/Favorites:** The Reader adds a manga to their favorites list or views their list. The system tracks their reading progress (last read page/chapter) so that it can be resumed later.
- **Comment on a Chapter:** The Reader leaves a comment on a chapter they have read and rates it. The system posts the comment (visible to others) and stores the rating. An extension of this use case is the system hiding or flagging inappropriate content, which involves an admin moderation scenario.
- **Share a Chapter:** The Reader shares a link to a chapter on a social platform (like sharing to Facebook). We discussed that this use case might simply generate a sharable URL or invoke a social media share dialog.
- **Request Uploader Role:** A Reader (who wants to upload content) submits a request to become an Uploader. The admin will then review and approve this request in the admin interface. Once approved, the user gains access to uploader functions.
- **Upload Manga Chapter (Uploader):** An Uploader user initiates a new chapter upload. This use case involves providing chapter info (metadata) and uploading images. After submission, the chapter waits for admin approval. A related sub-flow is **Edit/Update Draft** if the uploader wants to modify details before admin approval, or resubmit after rejection.
- **Approve/Reject Chapter (Admin):** Admin reviews a pending chapter submission. They can view the chapter details and either approve it (making it live) or reject it (sending feedback to the uploader). If rejected, the uploader could be allowed to make corrections and resubmit. This ensures quality control on content.
- **Manage Users (Admin):** Admin assigns roles (approve uploader requests) and can deactivate or suspend user accounts for violations. Also includes unlocking accounts or other user management tasks.

- **Feature Content (Admin):** Admin marks a chapter as "Highlighted"/"Recommended" to be featured on the homepage, with a timer for how long it stays featured.
- **View Analytics (Admin/Director):** Admin or Director views site analytics such as user statistics or revenue. This use case would involve the system querying aggregated data (page views, number of uploads, payments made, etc.) and displaying a report.

For each use case, we discussed basic flow and any alternate flows. These will be detailed in the SRS (Software Requirements Specification) document. We also noted any dependencies between use cases. For example, "Upload Chapter" depends on "Request Uploader Role" (only approved uploaders can upload), and "Translate Page" depends on the availability of the OCR/translation service.

6. Prototype Planning

As part of requirements analysis, the team agreed on creating **low-fidelity prototypes** for some critical interfaces. The goal is to visualize and validate the user experience for complex use cases. We identified the following screens to prototype:

- **Reader interface screens:** e.g., the **Manga Reader page** (showing a manga page with a translate button and navigation), and the **Home/Browsing page** (listing manga with filters).
- **Uploader interface screens:** e.g., **Upload form page** where an uploader enters chapter details and uploads files.
- **Admin interface screens:** e.g., **Pending Approvals list** (showing chapters awaiting approval) and an **Admin dashboard** snippet for user management.

These prototypes will be simple wireframes focusing on layout and available actions, not final designs. Nguyen Thanh Tien volunteered to start sketching these UI layouts. By having a prototype for each major use case, we can better verify if any requirement is missing or if any UI constraints affect requirements (for instance, realizing if additional data is needed to be captured). The prototypes will also be included in the requirements documentation as illustrations of expected system behavior.

7. Deliverables & Timeline

The team reviewed what deliverables need to be produced by the end of the Requirements Analysis phase and noted the deadlines according to our Gantt timeline:

- **Q&A Documentation:** A table of questions and answers from any requirements elicitation or stakeholder clarification. (Since this is a student project, this may include questions we anticipate and answers we derive from the project description/instructor feedback.)
- **Functional Requirements List:** A complete list of all functional requirements, each with a unique identifier and description. This will be part of the SRS document. We will organize them by categories (user account, reading, admin, etc.).
- **Non-Functional Requirements List:** A documented list of all NFRs (performance targets, security standards, etc.), also to be included in the SRS.
- **Use Case Catalog:** A section in the SRS detailing each use case (with brief descriptions or use case diagrams). This includes actors, preconditions, main flow, alternate flows for each key use case.
- **Prototypes/Wireframes:** Initial mockups for the selected screens, to be included in the appendix of the SRS (or as separate prototype document).
- **Updated Project Timeline:** Although not a deliverable per se, we will update the project schedule if needed based on any new insights (for example, if some requirements are more complex, allocate more time in design/implementation for them). The Gantt chart for the project already allocates roughly two weeks for this Requirements Analysis phase (mid-November to end of November). We are on track to complete the SRS by the end of November.

We confirmed the **deadline for the SRS document is November 30, 2025**, as per the course schedule. That gives us about two more weeks from this meeting date.

8. Action Items Assignment

Finally, responsibilities were assigned to ensure all tasks are covered in the coming days. The action items and owners are:

- **Nguyen Thanh Tien** – Lead the compilation of the **Functional Requirements** section of the SRS. Tien will consolidate the feature list discussed (in clear requirements wording) and ensure each requirement is clearly stated with acceptance criteria where possible. *Due:* Draft by Nov 7.
- **Tran Gia Huy** – Lead the **Non-Functional Requirements** documentation. Huy will write up the performance, security, and other NFRs, pulling in any necessary metrics from the discussion (e.g., response times, etc.). *Due:* Draft by Nov 22.

- **Nguyen Thanh Luan** – Develop the **Use Case descriptions**. Luan will write brief narratives for each use case identified and possibly draft a use case diagram. This includes outlining how each role interacts in each scenario. *Due:* Draft by Nov 7.
- **Le Anh Duy** – Create the **UI prototypes/wireframes** for the specified key screens. Tien will use a simple tool (e.g., Figma or even hand-drawn scanned sketches) to illustrate the Reader page, Upload form, and Admin approval page. *Due:* Initial prototypes by Nov 7, for review.
- **All Members** – Review and refine the **Q&A list** (we will collectively come up with any open questions to ask our instructor or things we need to clarify, e.g., "Do we need multi-language support beyond translation?", etc.) and fill in answers based on assumptions or instructor feedback.
- **All Members – Review each other's drafts**: On Nov 25, the team will meet informally to review the drafts of FRs, NFRs, use cases, and prototypes to ensure consistency and completeness.
- **Final SRS Assembly** – Le Anh Duy (as team lead) will integrate all sections into the final Software Requirements Specification document, with everyone proofreading the final version. *Due:* Nov 29 (to allow one day buffer before submission).

We also agreed to keep our communication active on Discord for any clarifications and hold short ad-hoc meetings if needed while working on these items.

Decisions Made

- The team **confirmed the project scope and objectives** as originally proposed – focusing on a manga platform with OCR-based translation. No scope changes were introduced in this meeting, meaning we will not add any new major features beyond those discussed (ensuring we can finish on time).
- We decided to **use third-party OCR and machine translation services** rather than developing our own OCR/translation algorithms. This decision was made to save development time and leverage existing reliable services (e.g., Google Vision API for OCR and Google Translate API or similar for translation).
- The **technology stack** for implementation was reaffirmed: **Next.js** (React framework) for the frontend and **Node.js + Express** for the backend, with a **PostgreSQL** database. This was initially set during the proposal, and at this meeting all members agreed these choices are suitable to meet our requirements.

- We agreed on the **three primary user roles** (Reader, Uploader, Admin) for the system. A potential "Director" role will not be a separate user type in the system but rather represented through admin privileges (for viewing stats). Essentially, any "director" functions (like viewing revenue) will be accessible to Admin users in the interface.
- The team set **performance targets** as part of the requirements (e.g., search <2s, page translation ~5s, page loads ~2s) and **security measures** (HTTPS, password encryption) which we will treat as binding requirements in design and implementation. These targets were agreed upon to ensure a good user experience and proper security baseline.
- We decided that certain **stretch features** (like a comprehensive analytics dashboard or multi-language UI) would be documented but marked as optional if time permits. Core features (reading, translating, uploading, approving, basic payment) take precedence. This decision ensures we focus effort on essentials first.
- The meeting established that **all requirement definitions will be finalized by Nov 25**, after which any changes should be minimal. This is to avoid scope creep in later phases. Any critical new requirements identified after this phase would need to be evaluated carefully against our timeline.

Action Items Summary

- **Nguyen Thanh Tien:** Draft Functional Requirements section of SRS (by 7/11/2025).
- **Tran Gia Huy:** Draft Non-Functional Requirements section of SRS (by 7/11/2025).
- **Nguyen Thanh Luan:** Write Use Case descriptions & diagram (by 7/11/2025).
- **Le Anh Duy:** Create UI prototype sketches for key screens (by 7/11/2025).
- **All Team Members:** Contribute to Q&A list for requirements clarifications; review and integrate all parts into final SRS (by 7/11/2025).
- **Next Meeting:** Schedule a follow-up meeting on **November 20, 2025** to commence the Software Design phase planning (this will directly follow the completion of the SRS documentation). Each member should come prepared with any design ideas or questions stemming from the requirements.