Project GameStudy Systems Report

1. Identifying and defining

1.1. Define and analyse problem requirements

Modern students struggle with time management, procrastination, and lack of motivation while studying. Traditional timers and study applications do not have sufficient engagement or motivation to support regular study routines.

The problem is divided into the following components:

- No motivation: Students become demotivated while using traditional timers.
- No reward system: There is no sense of progress or reward in traditional study tools.
- Limited accountability: Without competition or community, students tend to fall off track.
- Poor tracking: Users will typically have little access to useful study analytics or progress updates.

According to research, nearly 70% of students struggle to stay motivated during study sessions, which is typically caused by distractions and a lack of quick feedback. Traditional study timers fail to solve these issues because they lack engaging components that encourage good study habits. This gap emphasises the need for an application like GameStudy, which combines gamification and social accountability to promote long-term motivation and better study outcomes.

Needs and opportunities

Need	Description
1. User Engagement / Retention	The app must foster long-term student commitment by retaining and displaying individual progress data across sessions. This includes storing user profiles with study history, XP levels, and unlocked achievements while allowing seamless access through its Progressive Web App capability on any device. The UI must remain intuitive with simple navigation to facilitate daily use while offering an

	easy-to-view presentation of individual statistics and progress trends that encourage daily study habits.
2. Reliable Study Tracking	The timer should have the capability to begin, stop, and reset as well as track real-time elapsed time. Every session should generate secure and aggregate logs, accumulating comprehensive logs illustrating unique study sessions and aggregate totals over days, weeks, and months. The certified logs will automatically trigger the level progression and badge releases within the reward system based upon exhibited study effort.
3. Leaderboard	The app should include a competitive leaderboard to promote social responsibility and encouragement through low-stakes competition. This will display leading users by XP gained or study hours, thereby encouraging consistency and participation without becoming too distracting.
4. Data Security and Privacy	The system must have complete protections to safeguard all user data such as study logs, personal profiles, and social interaction. This necessitates secure access to accounts via encrypted authentication, password retrieval procedures with adequate protections, and role-based access to the data. All the data stored, such as study session information and achievement records, must utilize encryption when transmitted and stored and have adequate user controls over the sharing settings. The implementation must adhere to the standards of educational data protection while having clear privacy policies declaring the collection and use of the data.

GameStudy prioritises accessibility and usability to satisfy the functional needs of its clients, offering an accessible interface for a broad student population, including those with limited technical abilities. The Progressive Web App design is cross-device compatible, allowing users to access content seamlessly on smartphones, tablets, and desktop computers. Furthermore, the

gamified reward system adheres to educational psychology principles by offering rapid, clear feedback on progress, which promotes intrinsic motivation and sustained engagement.

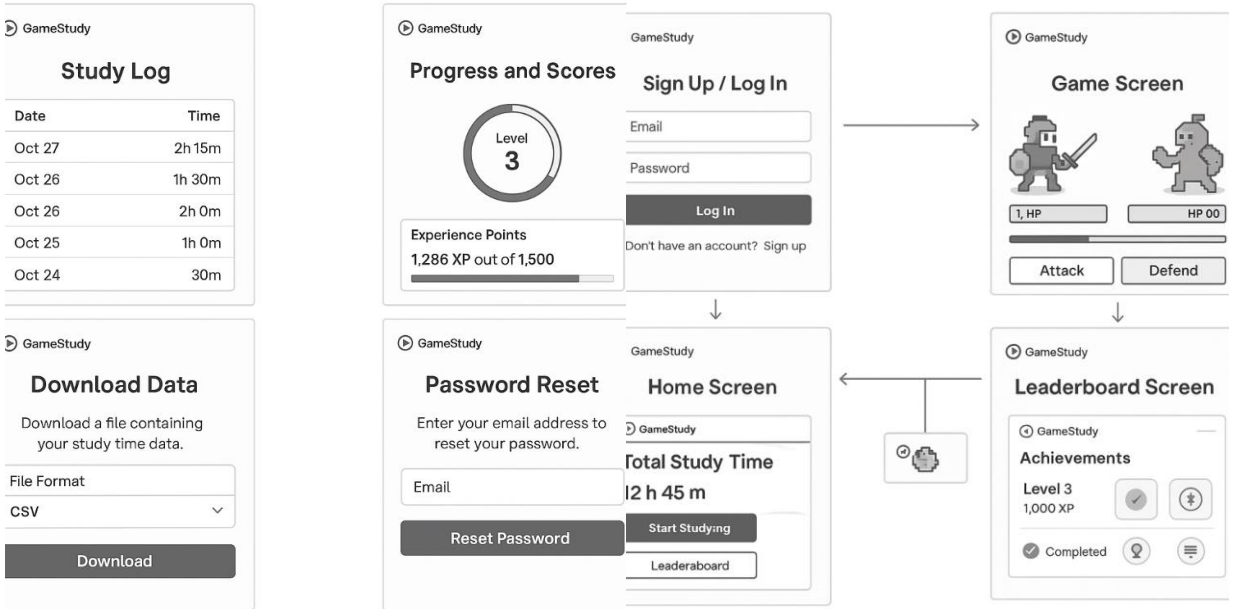
Boundaries

GameStudy will operate within a defined set of technical and practical boundaries to offer functionality, security, and accessibility. Being a Progressive Web App (PWA), it performs depending on how well the browsers perform and must rely on modern web standards to offer compatibility across leading browsers including Chrome and Safari. It will not operate well in older or non-supported browsers. Hardware-wise, the system must be supported on standard hardware such as smartphones, tablets, and desktop computers. But it can be different on low-end hardware with limited processing power or memory, and no functionality is planned for smartwatches, consoles, or other non-standard hardware.

Security is also a boundary. GameStudy will employ safe login, encryption storage of data, and MFA (multi-factor authentication), but it remains dependent on third-party platforms (e.g., Firebase) for backend data handling. This introduces reliance on their uptime, privacy policies, and infrastructure. The app will also need to adhere to data protection legislation (e.g., GDPR or Australia's Privacy Act) when collecting and storing users' data. Finally, there is a limitation with internet access: as an online web application, GameStudy will require a good internet connection to synchronize data, authenticate users, and provide updates.

1.2. Tools to develop ideas and generate solutions

Prototypes



1.3. Implementation method

For the GameStudy application, Phased Implementation would be optimal. Since GameStudy is a gamified productivity app for students, launching the system in phases allows sequential testing and improvement of key features. The first phase will show basic features like study time tracking and simple achievements, which are essential for user engagement. The following stages will build on this, adding more advanced features like leaderboards and social sharing, all on positive user feedback at each stage. In contrast to pilot implementation, where a few features are piloted with a small group, Phased Implementation allows us to roll out more extensively with continuous user interaction. This enables us to incrementally refine features, acting on user needs and ensuring each feature of the app works as intended. Phased Implementation is a more adaptable, iterative method, reducing the risk of releasing a faulty product but still enabling incremental introduction of extra functionality. Phased Implementation is also preferable to parallel implementation, which would be wasteful and for this case unnecessary, as there is no existing system to have in parallel operation. Direct cutover, while efficient, is too risky without iterative testing and feedback, which would result in possible issues after the full launch. Ultimately, Phased Implementation strikes the optimal balance between controlled feature rollouts, user-driven enhancements, and risk minimization and hence is the optimal way to deploy GameStudy.

1.4. Financial feasibility

Application Overview & Market Justification

Application Description

GameStudy is a gamification web application for effective study with students who have trouble maintaining a consistent study schedule. It tracks study time, builds habitual usage through virtual achievements and level ups, and introduces a friendly competition element through a leaderboard among friends.

Target Market

- High school and college students, 15–25 years of age

- Internet-savvy individuals with regular internet access mainly

- Worldwide reach, but initially targeting English-speaking regions like Australia, the UK, and the US

Unique Characteristics

- Gamified progression system (XP, levels, achievements)

- Competitive leaderboard to encourage consistent studying

- Simple and minimal UI for maximum focus

- Social accountability without distractions

Niche

There are many study aids and timer apps out there, but not many uses gamification and competition within an empty, distraction-free space. Most websites that exist either focus solely on tracking or lean far too heavily on social features which are distracting.

Demand vs Supply

Current study aid apps are either minimalist (timers-only) or overly complex. GameStudy bridges the gap for a casual, enjoyable, motivational app without unnecessary fluff.

Competitive Advantage

- Clean, intuitive interface that encourages use without distraction

- Freemium with cosmetics available for premium buy

- Cross-platform as a PWA (no reliance on app store)

- Building community via low-stakes competitive leaderboard

Projected Costs Statement

Item	Estimated Cost (AUD)
Domain registration (1 year)	$20
Web hosting (12 months)	$120
Developer time (200 hrs @ $40/hr)	$8,000
Client meetings (10 hrs @ $20/hr)	$200
Testing (manual on personal/test devices)	$50
Initial marketing (organic campaigns, social media)	$100
Miscellaneous costs	$100
Total Cost to Launch MVP	**$8,590**

Item	Estimated Cost (AUD)
Hosting and server maintenance	$180
Domain renewal	$20

Bug fixes and updates (50 hrs @ $20/hr)	$1,000
Marketing and community management	$500
Customer support (basic chatbot or self-service)	$100
Total Running Costs (Year 1)	**$1,800**

Projected Revenue (First Year)

Revenue Source	Estimated Income (AUD)
Premium subscriptions (500 users @ $30)	$15,000
Small school licenses (4 schools @ $1000 each)	$4,000
Ad revenue (banner/interstitial ads)	$1000
Total Estimated Revenue (Year 1)	**$20,000**

Market Survey Summary

Geographic Focus:

Primarily English-speaking countries with a strong student base and cultural familiarity with productivity apps.

Competitors:

- Forest App – Strong brand, focuses on staying off phones. Lack of gamified competition.
- StudySmarter – More of a study notes platform; not a gamified study tracker.
- Habitica – Broader habit tracking RPG app, not focused on studying.

GameStudy Advantages:

- Simple interface
- Niche focus on study hours, not general productivity
- Lightweight PWA access
- Friendly competition without pressure

Market Volume:

Estimated targetable student base: 50,000 (conservatively in a regional launch).

Estimated conversion to paying users: 2% = 1,000 users in year one.

Expansion Opportunities:

- Schools/educators promoting it
- Partnering with universities
- Expanding to group features or study communities

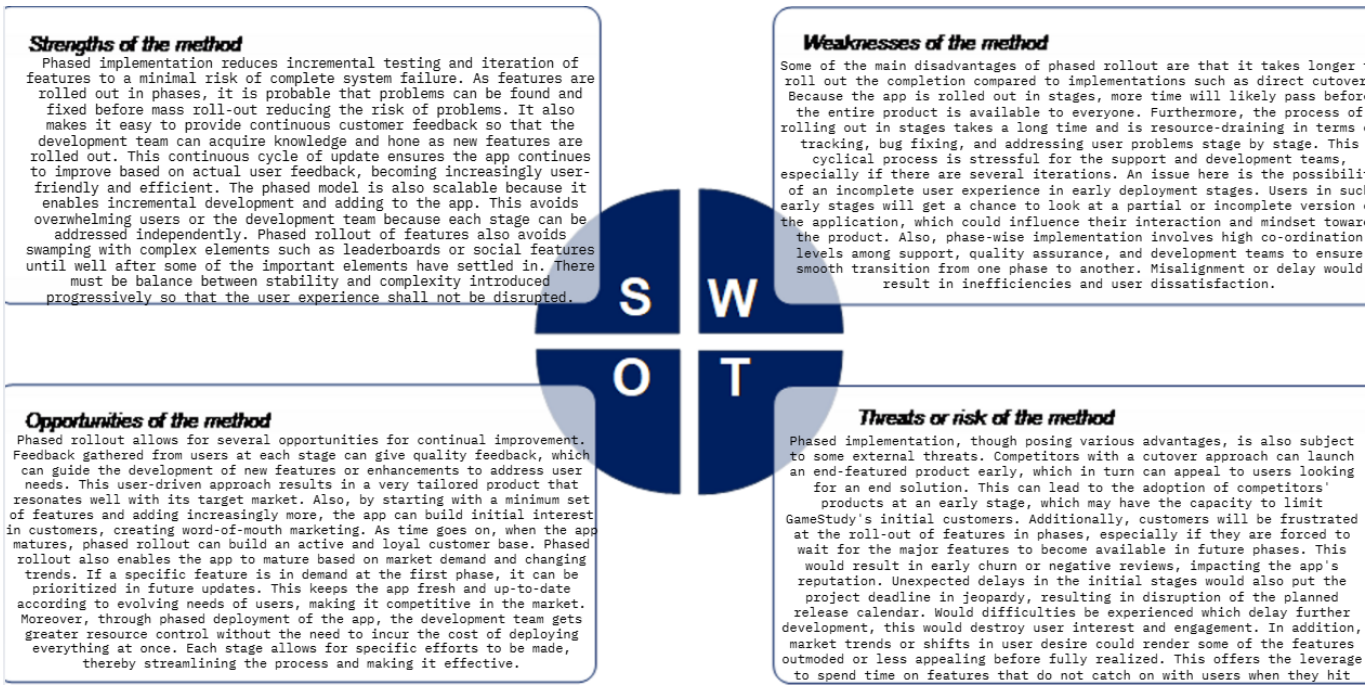
Review and Analysis

- Projected First-Year Profit:
Revenue ($20000) - Costs ($6390) = $13610

- Risk Factors:
 - User acquisition could be lower than expected
 - Platform fatigue in the productivity app space
 - Free tools may devalue premium features

- Mitigation Plans:
 - Focus on strong feedback from pilot implementation
 - Encourage word-of-mouth growth
 - Regular feature rollouts to maintain engagement

SWOT Analysis



Study	Go or No Go?	Assessment and evidence
Market feasibility	Go	There is strong demand from students for study habit-enhancing products. Gamification of edtech is increasing. Games like Forest and Habitica show that gamified productivity

		software is sellable and highly demanded, proof of market demand.
Cost of development	Go	Development will utilize open-source software (i.e., SQLite, Flask, and PWA frameworks). No proprietary software needs to be used. The primary investment is development time, which is within the budget of an individual or small group. MVP development cost is reasonable at $8,590 AUD, with most allocated to developer time. It is affordable for a small team or sole developer.
Cost of ownership	Go	First-year running costs are low at $1,800 AUD, making it sustainable even with modest user numbers. The hosting, domain, and database costs are relatively low for a lightweight PWA.
Income potential	Go	Monetization possibilities incorporate freemium features, such as paid themes or stat tracking, ad revenue, university and school licenses, or Patreon donations. Educational apps possess a consistent possibility for ongoing revenue. Projected revenue of $20,000 AUD in year one, from subscriptions, school licenses, and ads, creates a healthy $13,610 AUD profit margin.
Future expansion opportunities	Go	High potential to scale via educational partnerships, group features and school licensing. PWA structure also allows global reach with minimal platform Dependency. Can be extended to provide social features (study groups), school accounts, AI study coach, productivity tool integration (Google Calendar, Notion). App can scale with additional modules and user bases.

2. Research and planning

2.1 Social and Ethical Issues

Social and Ethical Issues – GameStudy Project

Social Issues:

GameStudy has significant social implications on three levels. At the individual level, the project enhances technical skill acquisition and employability in the technology sector while anticipating individual commitment to fairness through universal design choices such as screen reader accessibility and low-bandwidth capability for students of all socioeconomic levels. The development team must respond to environmental footprint in general through sustainable practices: minimizing code inefficiency to prevent digital waste, selecting green hosting for energy-friendly server deployment, and employing lean data storage processes. Cultural sensitivity leads at this team level to create adaptive learning systems.

To external stakeholders (institutions, teachers, and students), the creation of GameStudy creates actual social value in the form of a cheap alternative to motivational education software that is commercial, maybe restricting institutional costs and enabling more equitable access to educational software. The project also creates positive social effects in that it makes healthier study habits possible with its structure and creating indirect employment in app support, training, and technical support jobs created by its system. Through the development process, these social factors from the coding decisions of one individual to team and stakeholder influences on the world as a whole shape GameStudy's effect on society.

Ethical Issues:

Ethical development of GameStudy should be meticulously balanced in each stage of interaction. For the individual developer, this means prioritizing users' needs in terms of steering them away from manipulative design tactics like Duolingo's questionable streak system causing exhaustion and implementing Forest-type balanced usage notice in inducing good study practice. Developers also need to ensure accountability by using secure coding, transparent data processing like Quizlet's encryption methods, and monitoring algorithmic bias that harms some learning approaches. Ethical collaboration requires distinct processes for avoiding conflict of interest, particularly with monetization possibilities that cannot compromise educational integrity. The

groups need to have strict peer review systems and proper attribution systems (like GitHub's) and conduct diverse user testing so as not to have issues of bias.

To external parties, GameStudy needs to have strict health and safety systems in place, including features such as Forest's time limit on the screen and posture reminders for tracking physical well-being problems. The firm needs Duolingo-level data openness policy to institutions and schools but preserves full feature openness towards Apple levels to ensure complete equality for the disabled. Taking such a multidimensional ethical stance in preserving through all the developmental stages makes GameStudy not only socially sustainable but also educationally viable.

To summarise, the social and ethical considerations described above, shapes the development process of GameStudy. The considerations ensures that the application promotes equitable access, respects user privacy, and promotes healthy study habits. These principles are embedded through our explicit privacy policy which states that personal information may be collected such as your name, email address, and username when you sign up for the application.

you create an account.

2.2. Quality assurance

Quality criteria

Students **explain** quality criteria based upon the needs from Section 1.1. These quality criteria should contain qualities, characteristics or components that need to be included or visible – based on Section 1.1. – by the end of the current project.

Quality criteria	Explanation
User Authenticator	Implements Google Authenticator for 2FA alongside standard email/password login. Generates time-based one-time passwords (TOTP) with 30-second expiration. If 2FA is not available, SMTP verification is required.
Password Recovery	Recovery requires both email and 2FA confirmation from Google Authenticator. Temporary codes expire in 5 minutes.
Study Timer	Users can start, pause, and reset the study timer. The timer will track time spent studying in real time. Users will be able to view their total study hours for each session and accumulate them over time.

User Profile / Data Storage	The app will store user data through SQL, including study history, XP, achievements, and profile information.
Game Mechanics	Users will earn XP for each study session based on the amount of time they spend studying. Achievements and levels will be unlocked based on the cumulative study time. There will be leaderboards for users to compare their progress with peers or globally.

Compliance and legislative requirements

Students **explain** compliance and legislative requirements their projects need to meet and how they plan to mitigate them where possible. For example, projects that deal with sensitive personal data being publicly available may fall under the Australian NSW Privacy and Personal Information Act (1998) and/or Federal Privacy Act (1988). Alternatively, international standards on information security management such as ISO/IEC 27001 may also be applicable.

Compliance or legislative issue	Methods for mitigation
Privacy Act 1988 (Cth) and NSW Privacy and Personal Information Protection Act 1998	- Data Minimization: Gather only the essential user data (e.g. username, email for logging in). - User Consent: Display a clear privacy policy which users must consent to before using the app. - Access Control: Limit who has access to user data through authentication and role-based access - Anonymization: Keep study measures separate from any link to identifiable user data, where possible.
Spam Act 2003 (Cth)	- Opt-in system: Make users explicitly opt-in to receiving email updates, newsletters, or push notifications. - Unsubscribe functionality: Implement a working "unsubscribe" or notification toggle feature in settings. - Clear sender info: Make all communications easily identifiable by GameStudy branding and contact details.

	- Notification limits: Avoid spamming users with excessive messages or gamified push notifications.
Copyright Act 1968 (Cth)	- Use of Licensed Assets: Using media from royalty free sources and giving proper credit. - Creation of original content: Developing original icons, buttons, and branding whenever necessary. - Check licenses: Verify third-party materials meet the requirements of Creative Commons or commercial usage. - Document Usage: Keep a folder/log of all external assets and their licenses.

2.3. Systems modelling

Students are to develop the given tables and diagrams. Students should consult the Software Engineering Course Specifications guide should they require further detail, exemplars or information. Each subsection below should be completed with Section 1.1. in mind. Data dictionaries and data types. Students take the needs identified in Section 1.1. of this Systems Report. For each need, students identify the variables required, data types, format for display, and so on.

The data dictionaries provided directly address the main system requirements outlined in Section 1.1. User-related tables like Users, PasswordResets, and Logins ensure strong security and privacy measures and directly address Need 4 by managing authentication, password recovery, and session tracking securely. The StudySessions table records study activity, addressing Need 2 for reliable and accurate monitoring of user progress. Gamification features, for Need 1, are supported through the Achievements, Challenges, and UserDailyChallenges tables, which store information to motivate and engage users. Subject tracking and competitive features are managed via the Subjects table and related data structures, meeting Need 3's requirements for leaderboards.

Users Table

Variable	Data Type	Format for Display	Size in Bytes	Size for Display	Description	Example	Validation
id	Integer	Auto-increment	4	Up to 5 digits	Unique identifier for each user	101	Must be unique and positive
username	Text	Alphanumeric	50	Up to 20 chars	User's display name	studyhero1	Required, unique, 3–

							20 characters
email	Text	Email format	100	Email string	User's email address	user@email.com	Must match email format, unique
password	Text	Encrypted string	255	Hidden	Encrypted password	$2a$10$e.	Must be securely hashed
totp_secret	Text	Encrypted base32	255	Hidden	2FA secret key	JBSWY3DPEHPK3PXP	Required for 2FA if enabled
created_at	DateTime	YYYY-MM-DD HH:MM	8	DateTime	Account creation timestamp	2025-04-10 14:30	Must be valid timestamp
is_verified	Integer	Boolean (0 or 1)	1	0 or 1	Whether email is verified	1	0 = not verified, 1 = verified
total_study_time	Integer	Whole number	4	Up to 6 digits	Total study time in seconds	7200	Must be ≥ 0
xp	Integer	Whole number	4	Up to 5 digits	XP accumulated by user	1500	Must be ≥ 0
avatar	Text	Emoji or avatar	50	Emoji or text	User's selected avatar	😳	Optional, default = 😳

StudySessions Table

Variable	Data Type	Format for Display	Size in Bytes	Size for Display	Description	Example	Validation
id	Integer	Auto-increment	4	Up to 5 digits	Unique ID for each study session	201	Must be unique and positive
username	Text	Alphanumeric	50	Up to 20 chars	Username of user who studied	studyhero1	Must exist in Users
date	Date	YYYY-MM-DD	8	Date only	Date of the study session	2025-06-10	Must be valid date
seconds	Integer	Whole number	4	Up to 6 digits	Duration of session in seconds	3600	Must be ≥ 1

subject	Text	Free text	100	Up to 30 chars	Subject studied during session	Mathematics	Required, free input

Achievements Table

Variable	Data Type	Format for Display	Size in Bytes	Size for Display	Description	Example	Validation
id	Integer	Auto-increment	4	Up to 4 digits	Unique ID for each achievement	301	Must be unique and positive
username	Text	Alphanumeric	50	Up to 20 chars	Username of user who earned achievement	studyhero1	Must exist in Users
achievement_name	Text	Alphanumeric	100	Up to 30 chars	Name of the unlocked achievement	Study Streak: 7 days in a row	Required
unlocked_at	DateTime	YYYY-MM-DD HH:MM	8	DateTime	When the achievement was unlocked	2025-06-09 19:45	Must be valid timestamp

Challenges Table

Variable	Data Type	Format for Display	Size in Bytes	Size for Display	Description	Example	Validation
id	Integer	Auto-increment	4	Up to 5 digits	Unique ID for each challenge	401	Must be unique and positive
description	Text	Sentence	255	Up to 80 chars	Description of challenge	Study 2 hours in one session	Required
xp_reward	Integer	Whole number	4	Up to 4 digits	XP awarded upon completion	200	Must be ≥ 0

PasswordResets Table

Variable	Data Type	Format for Display	Size in Bytes	Size for Display	Description	Example	Validation

id	Integer	Auto-increment	4	Up to 5 digits	Unique ID for each reset token	501	Must be unique and positive
email	Text	Email format	100	Email string	Email of user requesting reset	user@email.com	Must exist in Users
token	Text	Alphanumeric	255	Hidden	Unique password reset token	8hGf9F3k9S...	Required, must be unique
expiration	Date	YYYY-MM-DD	8	Date only	Expiration date of reset token	2025-06-11	Must be future date

UserDailyChallenges Table

Variable	Data Type	Format for Display	Size in Bytes	Size for Display	Description	Example	Validation
id	Integer	Auto-increment	4	Up to 5 digits	Unique ID for each record	601	Must be unique and positive
user_id	Integer	Integer (FK)	4	Up to 5 digits	Linked user ID	101	Must exist in Users
challenge_id	Integer	Integer (FK)	4	Up to 5 digits	Linked challenge ID	401	Must exist in Challenges
date	Date	YYYY-MM-DD	8	Date only	Date the challenge is assigned	2025-06-10	Must be valid date
progress	Integer	Whole number	4	Up to 3 digits	Progress toward	80	Must be ≥ 0

					completion		
completed	Integer	Boolean (0 or 1)	1	0 or 1	Whether the challenge was completed	1	0 = not completed, 1 = completed

Subjects Table

Variable	Data Type	Format for Display	Size in Bytes	Size for Display	Description	Example	Validation
id	Integer	Auto-increment	4	Up to 5 digits	Unique ID for each subject	701	Must be unique and positive
username	Text	Alphanumeric	50	Up to 20 chars	Username owning the subject	studyhero1	Must exist in Users
subject_name	Text	Free text	100	Up to 30 chars	Name of the subject	Mathematics	Required

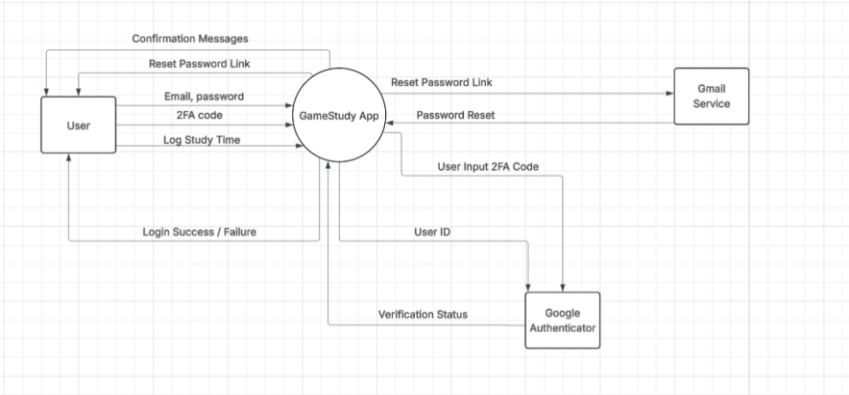
Logins Table

Variable	Data Type	Format for Display	Size in Bytes	Size for Display	Description	Example	Validation
id	Integer	Auto-increment	4	Up to 5 digits	Unique ID for each login event	801	Must be unique and positive
username	Text	Alphanumeric	50	Up to 20 chars	Username who logged in	studyhero1	Must exist in Users
login_time	DateTime	YYYY-MM-DD HH:MM	8	DateTime	Timestamp of login	2025-06-10 08:15	Must be valid timestamp

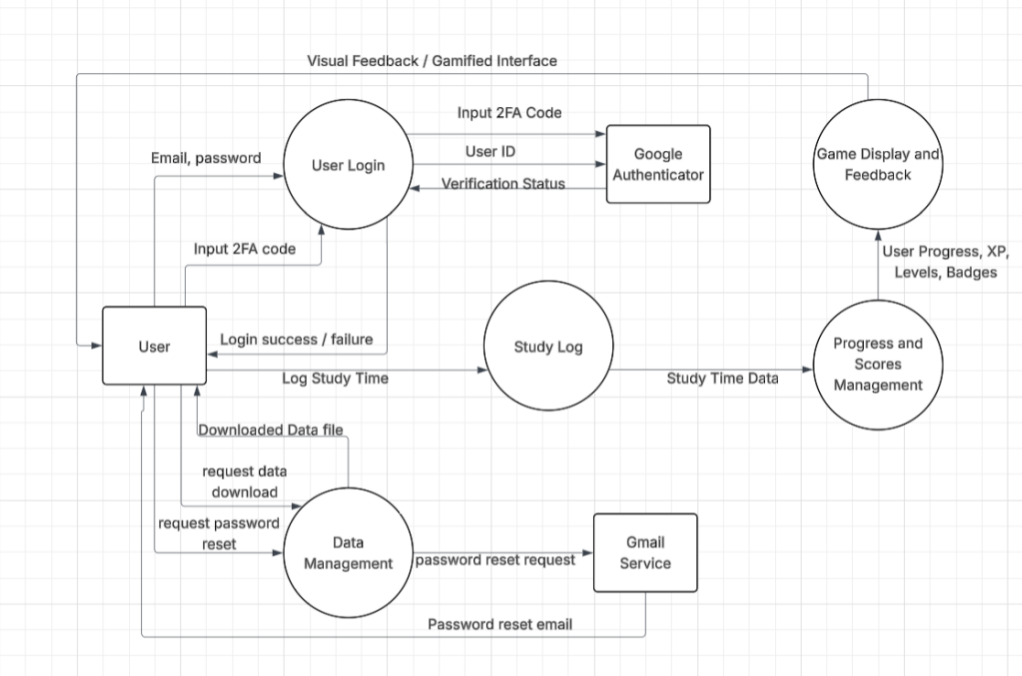
Data flow diagrams

Students develop data flow diagrams (DFDs) at Level 0 and Level 1. These diagrams should explicitly include the variables from the data dictionaries previously identified as well as the needs identified in Section 1.1.

Level 0

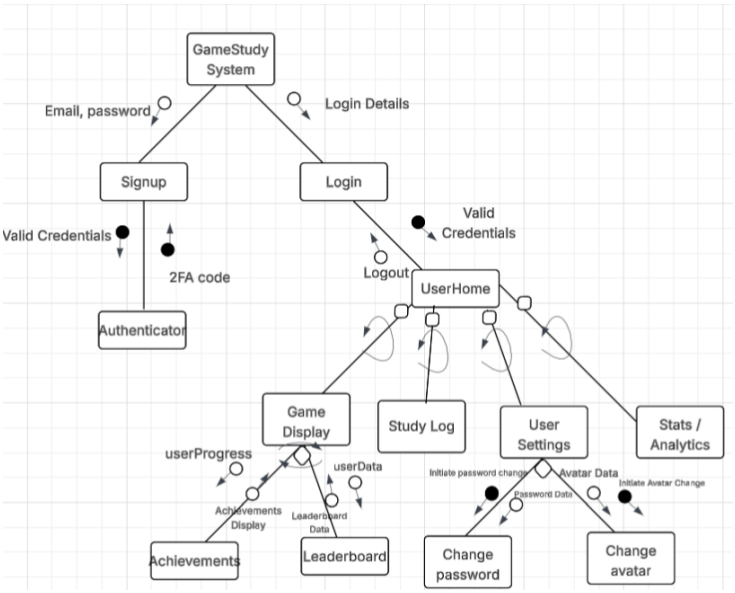


Level 1



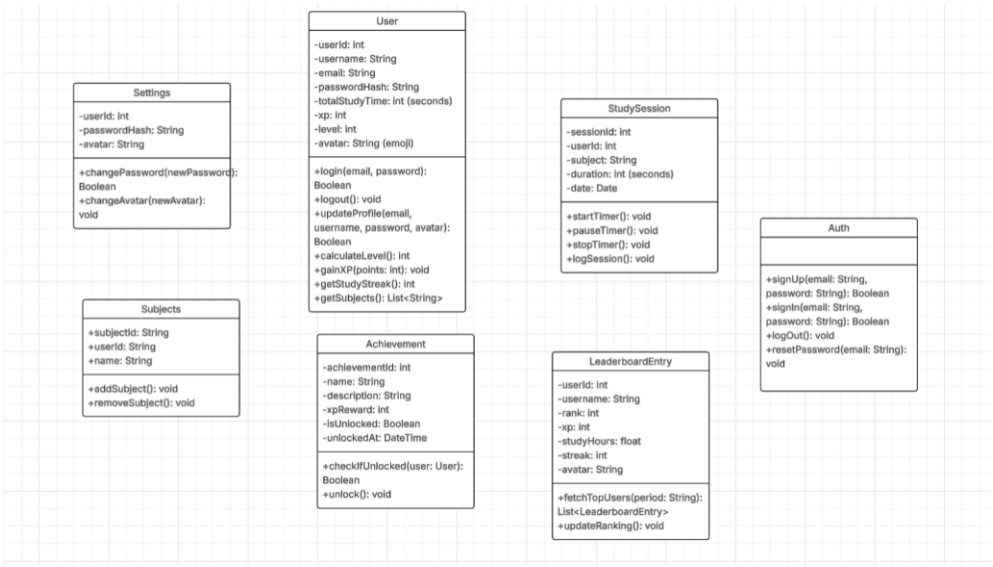
Structure charts

Students develop structure charts demonstrating how the procedures, modules or components of the final solution are interconnected.



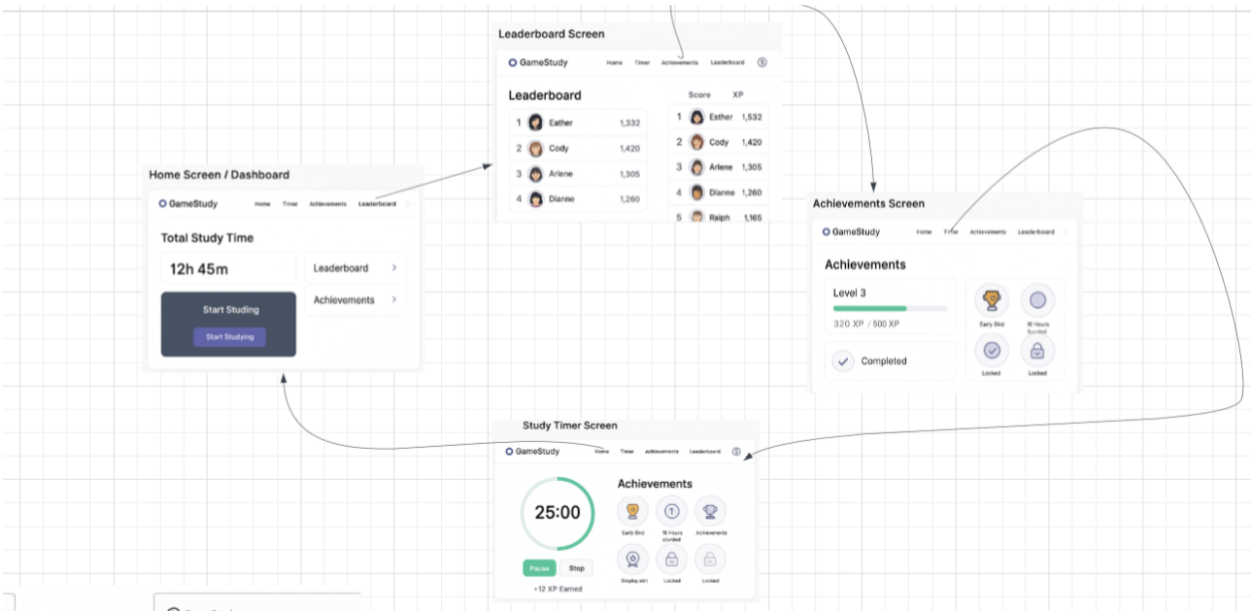
Class diagrams

Students develop class diagrams demonstrating how each class is related to the other.



Storyboards

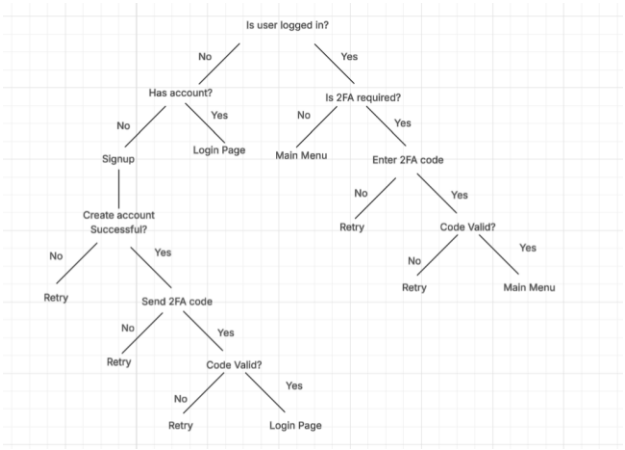
Students develop storyboards, visually representing the software solutions they will build.



Note: This storyboard includes the main user interaction screens: dashboard, study timer, achievements, and leaderboard. Authentication screens (signup, login, 2FA) are part of the system but are not shown here.

Decision trees

Students develop decision trees to visually outline the logic flow and chain of decisions or selections the final solution will need.



Algorithm design

Students develop algorithms using methods such as pseudocode or flowcharts to solve the problem and meet the needs from Section 1.1. These algorithms should explicitly include the variables from the data dictionaries created in the previous section.

```
BEGIN
  DISPLAY "Welcome to GameStudy!"
  LOOP
    DISPLAY "Choose: SIGNUP, LOGIN, EXIT"
    INPUT choice
    IF choice = "SIGNUP" THEN
      CALL Signup()
    ELSE IF choice = "LOGIN" THEN
      CALL Login()
    ELSE IF choice = "EXIT" THEN
      DISPLAY "Goodbye!"
      BREAK LOOP
    ELSE
      DISPLAY "Invalid choice, try again."
    ENDIF
  ENDLOOP
END
SUBROUTINE Signup()
  PROMPT and INPUT email, username, password
  VALIDATE inputs
  IF invalid THEN
    DISPLAY error and RETURN
  ENDIF
  HASH password
  GENERATE 2FA secret
  STORE user data (email, username, hashed password, 2FA secret)
  DISPLAY "Signup successful! Scan QR code for 2FA setup."
END
SUBROUTINE Login()
  PROMPT and INPUT username/email, password
  VERIFY credentials
  IF invalid THEN
    DISPLAY "Login failed" and RETURN
  ENDIF
  PROMPT and INPUT 2FA code
  VERIFY 2FA code
  IF invalid THEN
    DISPLAY "Invalid 2FA code" and RETURN
  ENDIF
  DISPLAY "Login successful."
END
SUBROUTINE StudySession()
  PROMPT "Start study timer? YES/NO"
  INPUT response
  IF response = "YES" THEN
    START timer
    DISPLAY instructions for PAUSE/STOP commands
    LOOP
      INPUT command
      IF command = "PAUSE" THEN
        PAUSE timer
      ELSE IF command = "STOP" THEN
        STOP timer
        RECORD session duration
        UPDATE user total study time and XP
        EXIT LOOP
      ELSE
        DISPLAY "Invalid command"
      ENDIF
    ENDLOOP
  ELSE
    DISPLAY "Session cancelled."
  ENDIF
END
```

Note: Only core flows being Signup, Login, and StudySession are included.

2.4. Project management

Software development approach

Agile methodology is the most suitable way of creating a competitive study-tracking Progressive Web App (PWA). Unlike the Waterfall approach, agile focuses more on flexibility, iterative improvement, and high-speed delivery of working software. The method is particularly suitable for high frequency updates and user feedback projects to enable the final product to be based upon user and market needs.

Agile development employs iterative two-to-four-week cycles in the form of sprints. Prioritized lists are released in each sprint with the aim of keeping the team active in progress while remaining flexible. The agile approach allows for consistent feedback to drive the app forward instead of rigid up-front requirements and specifications. By dividing the work into more manageable increments, the team can easily address issues, experiment with new ways of doing things, and change upon deployment without being limited to a deadline. The other key aspect of Agile is its emphasis on communication and interaction. Meetings each day ensure everyone involved is aware and in sync with the entire team, and sprint planning makes progress and priorities visible to all. This process is organized but flexible and provides momentum while also providing scope to introduce new requirements or unexpected blockers.

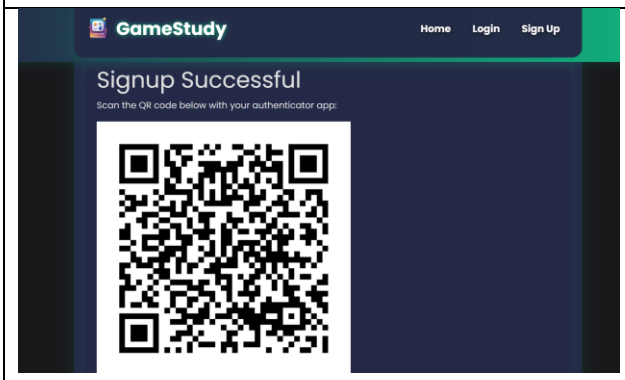
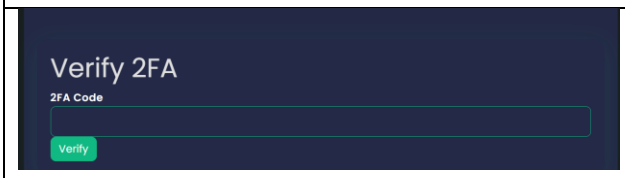
Agile also facilitates scalability and thus can be used in projects ranging from small to big ones. Basic principles can be used for small teams, and more structured models can become necessary while bigger projects need to be scaled up. The process can be used across a wide range of development work, including front-end work, back-end work, and quality assurance work. Using Agile principles as part of the development process, teams can deliver quality software within deadlines and respond to user needs and requirements and successfully compete in fast and dynamic markets. It makes the study-tracking PWA dynamic, user-friendly, and capable of evolving and developing new functionalities as and when needed, leading to a more successful and sustainable product. Project management technologies such as Gantt charts and GitHub branches will be used for task tracking, sprint planning, and problem management. User and client input gathered throughout each sprint will be thoroughly analyzed during sprint retrospectives to priorities improvements and new feature requests. This constant feedback loop ensures that the product evolves according to user requirements while maintaining excellent quality.

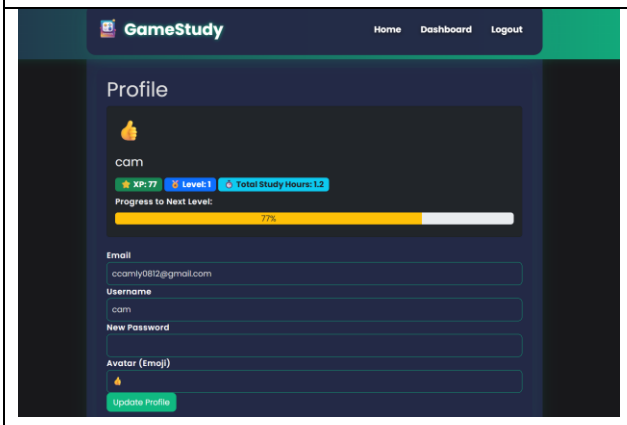
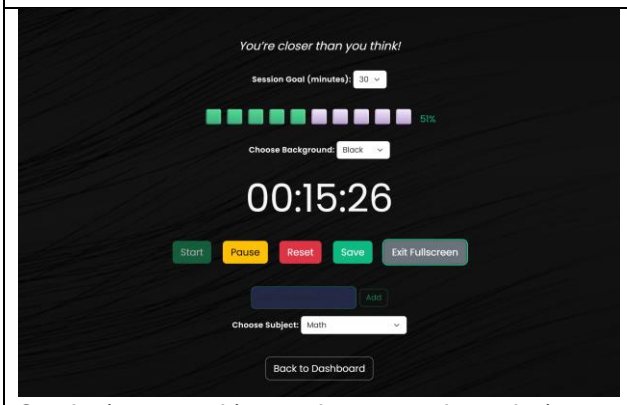
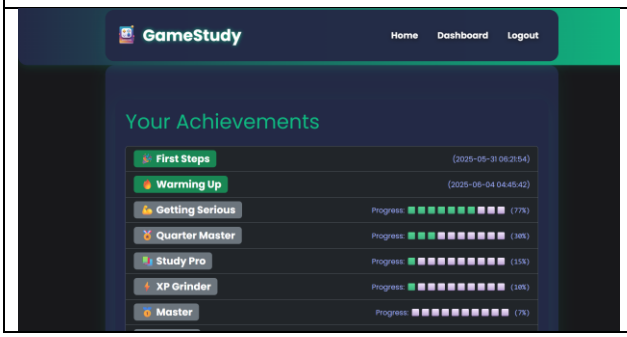
3. Producing and implementing

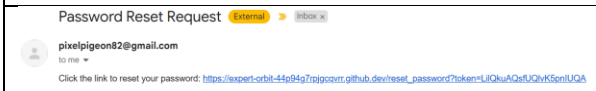
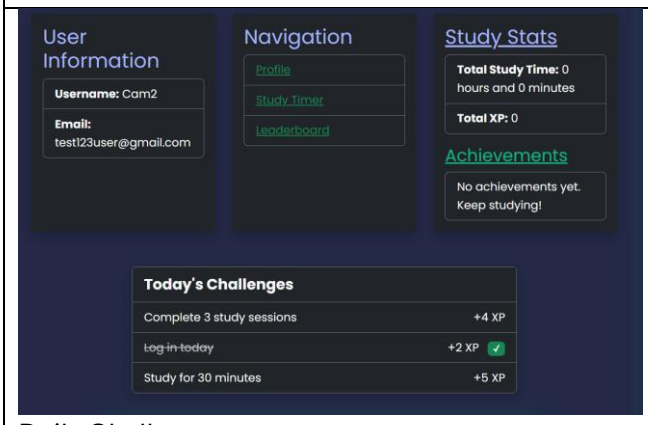
Solution to software problem

Students is to **include** screen shots of their final developed solution here. Each screenshot should include a caption that **explains** how it links to the:

- Needs identified in Section 1.1.
- Components of Section 2.3. such as the storyboards, data dictionaries and so on.

Screenshots	Explanation
 QR Code for 2FA	By providing a Google Authenticator QR code, the app enables users to create time-based one-time passwords (TOTP) with a 30-second renewal. This configuration ensures that only the rightful owner of the account can access the system because authentication is now based not only on a password but also on the physical device where the authenticator resides. This feature specifically addresses Need 4: Data Security and Privacy because it strongly resists unauthorized users accessing users' data, including profile information, study sessions, and accomplishments. Besides this, it also addresses 2.3 User Authenticator by employing TOTP-based two-factor authentication with a safe and encoded login system in accordance with modern cybersecurity standards in learning systems.
 Verify 2FA code	Requiring users to enter the 6-digit TOTP from their authenticator app adds a second layer of security. This measure significantly reduces the risk of account compromise, especially in cases of stolen or leaked credentials. This provides users with protection from common attack vectors like brute-force and phishing attacks so that their stored data, such as personal information, stays private. This also complies with Need 4: Data Security and Privacy but also complies with the User

	Authenticator specifications in Section 2.3, which mandates two-factor authentication via time-based codes.
 User profile with password and avatar change function	The ability to change and reset passwords and customize avatars keeps users in control of how the site looks and feels to them while also protecting account security and increasing engagement. Technically, password changing greatly enhances data security by allowing users to actively take control of their security issues. As such, this feature is critical in addressing Need 1: User Engagement / Retention, in terms of providing customization and control, and Need 4: Data Security and Privacy in terms of ensuring passwords are handled securely. It also improves the quality attribute of User Profile / Data Storage by securely storing users' data using SQL, as well as encrypting the data as and when required.
 Study timer tracking and constantly updating page through JS	When students interact with the study timer by starting, pausing, and continuing working on a task, they get surrounded by a dynamic interface that monitors their activities in real-time. The constant surveillance generates complex records that help compute XP, grant achievements and accomplishments, and provide them with a study of history. The app provides evidence of improvement and motivates the user to continue with study sessions. This specifically satisfies Need 2: Reliable Study Monitoring by providing users with a review of their session statistics and cumulative study time. Additionally, it meets the Study Timer need as specified in Section 2.3 with precise real-time measurement and the production of confirmed study records.
	By showing users their current achievements and their progress in unlocking new ones, the app creates a visible, gamified feedback loop. This feedback loop rewards users for consistent effort and encourages longer and more frequent study sessions. The effect is that users become more engaged and motivated, as they can visually track their performance and feel a sense of

Achievements page with achievement progress	accomplishment. This directly addresses Need 1: User Engagement / Retention by incentivizing regular app use. Additionally, it satisfies the Game Mechanics criteria in Section 2.3, which includes XP and achievement of systems based on time studied.
 Leaderboard page with podium system based off all time, this week and today	Users are ranked based on their XP or study time in different time intervals, today, this week, or all-time. This ranking creates a spirit of competition that sparks study as users are convinced to log in and study hard as they want to retain or improve their positions, especially if they are on the podiums. The result is an increased sense of community and responsibility, specifically meeting Need 3: Leaderboard. Further, the external recognition and rewards contribute substantially to User Engagement and Retention (Need 1). This corresponds with the Game Mechanics demand stated in section 2.3 in which social comparison and levels of XP determine long-term retention.
 Email for password reset request	When a user forgets their password and initiates a recovery, the app sends a secure reset link via email. The email is fetched from the database for the username or email that was entered so only the owner can change credentials. This supports Need 4: Data Security and Privacy by putting strict access controls in place. Moreover, this also aligns with the Password Recovery quality criterion, where double-factor authentication and time-limited reset codes are needed to protect user data.
 Daily Challenges	The Daily Challenges feature adds dynamic, rotating objectives that are updated every 24 hours. These challenges encourage users to interact with the app daily by awarding bonus XP upon completion. This approach establishes short-term goals that encourage regular logins and study sessions, hence enhancing user retention. This feature addresses Need 1: User Engagement / Retention by providing a gamified structure that gives users a reason to return frequently, beyond simply tracking time. The rotation of challenges prevents

	stagnation and encourages the development of positive, habitual study behaviours. The Daily Challenges system further expands on the Game Mechanics mentioned in Section 2.3, providing greater depth to XP progression and achievement attainment. It was not initially specified but was recommended as an additional option.

As shown in the photos and explanations above, the project not only fits all the basic client needs, but it also incorporates several more inventive solutions. These include the Stats Page, a podium-based Leaderboard system with daily, weekly, and lifetime ranks, as well as the Daily Challenges feature. These were not part of the initial requirements, but were proposed and implemented to improve user engagement, motivation, and long-term retention.

Version control

Students **describe** what version control system or protocol was implemented.

Version control was managed via Git with GitHub as a means of managing the remote repository. Changes were tracked with commit messages (e.g. adding leaderboard). I used a sprint branching system, building each sprint or big feature out in its own branch (e.g., sprint2 for UI and study session tracking) and merging into main via pull requests. This enabled organized development with easier bug tracking and clean integration. It was used as a mini developer's diary where the development was followed through sprints. Even though there was a proper developer diary that was being filled out, planning and task management was also done using GitHub issues and milestones. This consisted of traceability, versioning safety, and structured project management across the development cycle.

4. Testing and evaluating

4.1. Evaluation of code

Methodology to test and evaluate code

Students **explain** the methodologies used to test and evaluate code. Methodologies include:

- Unit, subsystem and system testing

- Black, white and grey box testing
- Quality assurance.

Unit testing is the process of testing functions or components individually, to ensure that each function works as expected. Unit testing was used to ensure that essential functions such as XP generation, password hashing and form validation worked as expected. For example, the function for converting study time to XP was tested using a range of values, including edge cases such as 0 and longer periods of time. Password check functions were evaluated to ensure that any invalid or bad passwords were recognised and rejected before they entered the database. The desired outcome of each unit test was that the function will generate proper, predictable outcomes for certain inputs, supported with error handling procedures for incorrect situations.

Subsystem testing ensures that modules inside integrating components operate in sync. In GameStudy, this covered processes such as login, session setup, and dashboard access. For example, after a successful login, the system will save a session token and automatically redirect the user to their dashboard, where they can view their experience points and progress. Subsystem testing was performed to guarantee that session management and database activities all functioned properly.

System testing thoroughly tested the application as a whole. Once the core features, such as authentication, timers, XP tracking, and leaderboards, were implemented, a thorough system test was performed to simulate real-world usage. A user would sign up, log their study time, make it through levels, and compare their statistics with other users. The purpose of this test was to ensure smooth navigation, free of any functional dead ends. Whenever inconsistencies were found, like wrong XP values, or redundant entries on the leaderboard, bugs were carefully documented and fixed in the following sprint.

Black box testing is a method where the internal workings, being the code, of the application are not visible to the tester; only the inputs and outputs are evaluated. It assessed frontend forms and interactions by entering valid/invalid data into signup, login, and timer fields. For example, submitting a login form with invalid password requirements should produce an error message, while correct credentials should log the user in. The goal was to simulate how an average user would interact with the app and ensure it responded properly under every circumstance. This was used to ensure proper input validation and that the frontend worked as expected.

White box testing involves a comprehensive investigation of the internal code paths and logic embedded within the application. As the developer developed the application, this strategy was used to detect logical bugs and remove unnecessary code paths. With full visibility into code paths, particularly in auth_routes.py and api_routes.py, logic errors and inefficiencies were identified using print statements and line-by-line debugging. Functions involved in session verification, backend routing, and form data management were subjected to a line-by-line review to ensure

their correct operation. The expected result was that each logical branch would work as intended for all inputs.

Grey box testing utilised both approaches of white box and black box testing. For example, XP accumulation based on timer duration was validated by cross-checking the frontend presentation to the backend status, which helped identify any inconsistencies or misaligned expectations. If a user studied for 30 minutes and earned 30 XP, the amount should be appropriately stored, represented in the database, and displayed on the user's dashboard and leaderboard. Any input validation, session management, and CSRF protection were enforced, and secure password storage was developed using bcrypt.

Quality assurance practices were also integrated into the testing process, this included:

- Code reviews were conducted to identify logic errors and improve code clarity.
- Input validation was implemented to both the frontend and backend to prevent incorrect data and reduce the risk of code injection or other potential security issues.
- The code had also included strong security protections such as CSRF tokens, session expiration limits, and bcrypt-based password hashing.
- Responsive design testing was performed by using browser developer tools to simulate different screen resolutions, ensuring accessibility and optimal layout on desktop, tablet, and mobile devices.

Code optimisation

Students **explain** the methodologies used to optimise code so that it runs faster and more efficiently. Methodologies include:

- Dead code elimination
- Code movement
- Strength reduction
- Common sub-expression elimination
- Compile time evaluation – constant folding and constant propagation
- Refactoring

Dead code elimination was used to remove unused variables and functions left over from early prototypes, particularly within authentication and utility modules, reducing unnecessary space and improving code readability. Code movement was applied in the main application file to relocate core setup functions such as CSRF protection, Flask-Mail, and session configuration to the top of the file, allowing for faster initialisation and clearer separation of concerns. For frequently executed logic, such as XP gain, study time conversion, and leaderboard updates, strength reduction was used by replacing complex conditionals and operations with simpler values, improving response time.

Common subexpression elimination was applied by factoring out repeated expressions in study tracking and achievement logic into shared helper functions, reducing redundancy and centralising logic for easier maintenance. Static configuration values like session timeouts, XP thresholds, and rate limits were set using constant folding and propagation, allowing these values to be declared once and referenced throughout the codebase without recalculating. Refactoring was used to improve modularity and maintainability, where repeated UI components were templated using layout.html, and backend processes like input validation, email formatting, and authentication checks were placed into reusable functions. These optimisations ensured that the application not only performed smoothly under use but also remained easy to extend and develop in later phases.

4.2. Evaluation of solution

Analysis of feedback

Students **analyse** feedback given to them on the new system they have just created. This feedback can be in the form of an interview, survey, focus group, observation or any other applicable method. Students should also include overall positive, negative or neutral sentiments towards the new system in their response.

User feedback was gathered through Google Form surveys and informal testing sessions with students from the target audience. The overall sentiment of the application was strongly positive. Users enjoyed the achievement system and leaderboard, noting that these features significantly enhanced motivation and made the study process feel rewarding and competitive. Several students commented on how the interface was clean, user-friendly, and easy to navigate, allowing them to focus on their study sessions without any distractions. No functional issues were detected

during user testing, and major functions such as user authentication, the study timer, and leaderboard updates worked properly. Minor suggestions were raised, including the introduction of additional visual themes, profile customization, and more detailed study statistics. While these features would enhance personalization and user satisfaction, they were considered non-essential to the core operation of the application. This feedback validates that the system met its intended purpose and was well received by the target audience. The positive reception of the gamification elements, ease of use, and reliable performance confirm the application's success. The suggestions provided also inform potential areas for future enhancement.

Testing methods

Students **identify** the method or methods of testing used in this current project. For each they use, students are to **explain** how and why it was used.

Method	Applicability	Reasoning
Functional testing	- Login, signup, timer, XP system, leaderboards	- Functional testing was used to verify that each feature performed correctly according to the requirements. Authentication, timer accuracy, and leaderboard updates were tested rigorously as they were core features of the application.
User Acceptance testing	- Final system as used by target students	- UAT was performed by having students use the application in realistic study scenarios to confirm that the system met their practical needs and expectations. -

Live data	- User input, session tracking, leaderboard entries	- Live data testing validated that real user data (such as study sessions, login attempts, and XP updates) was processed correctly and reflected accurately in the system.
Simulated data	- Invalid inputs, edge cases	- Simulated data was used to test how the system handled edge cases, invalid form submissions, and rapid session transitions without crashing.
Beta testing	- Pre-release for small user group	- Beta testing allowed a small group of early users to interact with the system and provide feedback on usability, bugs, and general user experience before finalizing.

Security Assessment

Students are to **perform** an extensive security assessment of their final application and e**xplain** the countermeasures implemented.

Threat	Countermeasure
SQL Injection	- SQL Injection, where attackers inset malicious SQL into input fields to manipulate database queries, and thus databases, is

	mitigated in GameStudy through the use of parameters that strictly treat all user input as raw data rather than executable code. This prevents malicious SQL from being misinterpreted as commands. Server-side input validation is also utilised to prohibit certain characters such as semi colons and SQL operators, which prevents attacks to the query processor.
Cross Site Scripting (XSS)	- Cross site scripting occurs when attackers inject malicious scripts into web pages that execute in the user's browser. This is mitigated in GameStudy by applying input sanitisation on all user inputs, specifically username, password and email, preventing characters such as <, >, ", to entry or are encoded before being processed. Sanitisation is enforced at both data entry and when rendering to the browser, preventing any chance for XSS.
Cross Site Request Forgery (CSRF)	- Cross Site Request Forgery (CSRF) attacks trick authenticated users into unknowingly submitting malicious requests. GameStudy mitigates this by embedding unique CSRF tokens in all form submissions through Flask-WTF. The sever verifies the token with the active session, making it impossible for attacks to forge or guess valid tokens. This ensures that all user actions are intentional and secures operations like logins, profile updates, and XP tracking from unauthorised users.
Invalid Forwarding and Redirecting	- Invalid forwarding attacks manipulate URLS and redirect users to malicious websites without their knowledge. GameStudy prevents this by limiting all redirects to hardcoded, verified paths, shown in the code as SAFE_DIRECT_URLS. This disables all other redirect parameters, which ensures that attacks cannot hijack navigation or trick users into visiting malicious external websites, protecting users from phishing and malware attempts.

Broken Authentication	- Broken authentication, where attackers exploit weak login processes to bypass account security, is mitigated in GameStudy by using bcrypt with salting to hash passwords. This makes even leaked credentials computationally impossible to crack through brute-force. Sessions are automatically set to expire after 30 minutes of inactivity, closing the window for session hijacking attempts. Together, these controls ensure password integrity, session confidentiality, and prevent unauthorised account access, even if some credentials are leaked.
Session Mismanagement	- Session mismanagement allows attackers to hijack sessions due to weak session handling. GameStudy prevents this by generating strong, unpredictable session keys, enforcing automatic session expiry, and fully clearing session data on logout. Session IDs are also regenerated upon each login to prevent fixation. These controls prevent token reuse and protect unauthorised account access by blocking attackers from exploiting predictable sessions.
Race Conditions	- Race conditions occur when simultaneous processes modify shared data, causing conflicts or system errors. GameStudy prevents this by applying strict database transaction management for XP updates, study time logging, and leaderboard calculations. Transactions ensure that database operations are executed sequentially and locked correctly, preventing users from overwriting each other's data or creating inconsistent records. This preserves system accuracy, especially under high user amounts during competitive periods.

Test data tables

Students **identify** variables which were used for either path and/or boundary testing. Students

develop these test data tables based on their algorithms versus their real code. Students then **state** the reason for including said variables.

Variable	Maximum	Minimum	Default Value	Expected Output	Actual Output	Reason for Inclusion
Username Input	30 characters	3 characters	Empty string	Accept input up to 30 characters	Accepts max 30 characters	- To correctly enforces maximum username length to prevent database overflows and layout issues.
Username Input	Special symbols (!@#$%)	Only letters	Empty string	Reject input with special symbols	Rejects special symbols	- To verify input sanitisation correctly blocks invalid characters, which could break display or introduce security risks.
Password Input	12 characters	8 characters	Empty string	Accept passwords 8 – 12 characters	Accepts min and max length	- To confirm that password length restrictions are correctly implemented, ensuring system security and user accessibility.
Password Input	Common word ("password")	N/A	Empty string	Reject common weak passwords	Rejects weak passwords	- To ensure only strong passwords are registered by rejecting weak ones, reducing the chance of unauthorised access.
Study Time Input	Negative values	0	0	Reject negative study time	Negative times rejected	- To ensure negative time entries are rejected, which prevents system logic errors and possible reverse time exploits.

Redirect Path input	Valid internal path	External URL	N/A	Allow only valid internal redirects	Blocks all external redirects	- To verify that redirects are restricted to safe, internal paths only, preventing URL manipulation and phishing attacks.
Leaderboard Position	1st	nth	N/A	Correct sorting of leaderboard positions	Correctly sorts users by XP	- To ensure that the leaderboard accurately sorts users based on XP in real-time, maintaining competitive fairness.

Boundary testing

Variable	**Boundary Value Tested**	**Expected Output**	**Actual Output**	**Reason for Inclusion**
Username Input Length	30 characters	Input accepted	Input accepted	Tests the upper boundary of allowed username length to ensure the system handles max input correctly.
Username Input Length	31 characters	Input rejected	Input rejected	Confirms the system correctly blocks inputs that exceed the maximum allowed limit.
Username Input Length	3 characters	Input accepted	Input accepted	Tests the lower boundary to ensure the system accepts the minimum username length as valid.
Username Input Length	2 characters	Input rejected	Input rejected	Ensures username validation correctly enforces minimum length requirement for security.

Password Input Length	12 characters	Input accepted	Input accepted	Tests the upper boundary of allowed password length to ensure the system handles max input correctly.
Password Input Length	13 characters	Input rejected	Input rejected	Confirms the system correctly blocks inputs that exceed the maximum allowed limit.
Password Input Length	8 characters	Input accepted	Input accepted	Tests the lower boundary to ensure the system accepts the minimum password length as valid.
Password Input Length	7 characters	Input rejected	Input rejected	Ensures password validation correctly enforces minimum length requirement for security.

Analysis of solution against quality success criteria

Students are to take each quality success criteria from Section 2.2 and place it here. For each quality criteria, **analyse** the components of the solution that met or did not meet each quality criteria. Give reasons why each success criteria were or were not met.

Quality criteria	Met?	Analysis
User Authenticator	Yes	The authentication system successfully implements both standard email/password login and Google Authenticator-based two-factor authentication (2FA). TOTP codes are securely generated and validated on the backend, adding an essential layer of protection against unauthorised access. The backup SMTP verification ensures users can still log in if 2FA is

		unavailable, maintaining usability without compromising security. All authentication components passed functional testing and security validation.
Password Recovery	Yes	Password recovery was built using the email address used to register the account. Temporary recovery codes expire after five minutes, greatly reducing the possibility of misuse. This recovery path has been successfully tested with several simulated user accounts to ensure functionality and security, meeting both user needs and security standards.
Study Timer	Yes	The study timer allows users to start, pause, and reset study sessions in real time. Accurate time tracking is maintained across sessions and device refreshes using session storage and backend updates. Functional testing confirmed that accumulated study time is correctly logged and displayed on user profiles, satisfying the requirement for real-time, continuous tracking.
User Profile / Data Storage	Yes	The user profile system successfully stores individual study histories, XP, achievements, and account details using a secure SQL database structure. Passwords are hashed using bcrypt, and all profile fields are sanitized to avoid SQL injection. The data retrieval and update operations were carefully evaluated and proved consistent performance across multiple user scenarios.
Game Mechanics	Yes	The gamified system accurately calculates XP based on study time and correctly unlocks achievements and levels. The leaderboard updates constantly, allowing users to compare their progress in real time. The XP and achievement logic were validated using unit tests and user input, confirming the system's proper interaction with the primary study functions. The game mechanisms meet both functional and user engagement success criteria.