Hands UP

Software Requirements Specification

Updated: 16/10/2025

Contents

1 Introduction

Hands UP is an innovative application that bridges the communication gap between signers and non-signers. Using advanced AI technology, the application detects and translates sign language in real-time through the device's camera, converting signs into both text and spoken language without significant delays. Beyond translation, it also serves as an interactive learning platform with structured lessons and feedback on signing accuracy.

2 User Stories

Note: A **user** *refers to anyone using the application, while a* **learner** *specifically refers to someone who has created an account.*

2.1 Registration

User Story: As a user, I want to create an account with my name, surname, username, email address and password so that I can access the sign language learning curriculum and additional app features.

Acceptance Criteria:

- All input fields are required
- Email address is validated for correct format
- Password must be at least 8 characters and include a special character
- Duplicate usernames and emails are not allowed
- Terms and Conditions are accepted
- Appropriate error message is shown for invalid sign up attempts

Definition of Done: My account has been created successfully and I am automatically logged in and redirected to the app homepage.

2.2 Login

User Story: As a learner, I want to login into my account using my email and password so that I can continue with the sign language learning curriculum or access additional app features.

Acceptance Criteria:

- All input fields are required
- User is authenticated against stored credentials
- Appropriate error message is shown for invalid login attempts

Definition of Done: I have successfully logged in and automatically redirected to the app homepage.

2.3 Forgot Password

User Story: As a learner, I want to reset my password via a link sent to my email so that I can regain access to my account if I have forgotten my password.

Acceptance Criteria:

- A reset link is sent to the email address if it exists
- Link redirects to a secure page to set a new password
- User receives confirmation of password change

Definition of Done: My password is updated and I can now log in with the new credentials.

2.4 Help

User Story: As a learner, I want to access a help section with guides, FAQs and contact support options so that I can quickly resolve any issues or questions I have while using the app.

Acceptance Criteria:

- Help section contains searchable guides, FAQs and troubleshooting tips
- Clear contact support option is available
- Help content is accessible from all screens
- All help content loads without errors and is easy to navigate

Definition of Done: I can easily access the help section, browse topics and contact support when needed without encountering errors.

2.5 Manage Profile Details

User Story: As a learner, I want to update my name, surname, username, email and password so that my account information remains valid and up to date.

Acceptance Criteria:

- All fields are editable
- Input is validated before saving
- Confirmation message is shown after successful updating

Definition of Done: My changes are saved and I can see the updated details.

2.6 Profile Picture

User Story: As a learner, I want to upload or change my profile picture so that I can personalise my account and make it visually identifiable.

Acceptance Criteria:

- User can upload images in supported formats (JPG, PNG)
- User can change or remove profile picture at any time
- Updated profile picture is displayed immediately after saving

Definition of Done: My uploaded or changed profile picture appears correctly on my profile and remains updated after I refresh or log back in.

2.7 Terms & Conditions

User Story: As a learner, I want to view the full terms and conditions of the platform, including how my data is collected, used and protected so that I understand my rights, responsibilities and the rules for using the app.

Acceptance Criteria:

- Terms & Conditions are viewable in a dedicated section
- Link to Terms & Conditions is available at sign up and in settings
- Content matches the most recent approved version

Definition of Done: I can open and read the complete Terms & Conditions, knowing they are up to date and accurate.

2.8 Logout

User Story: As a learner, I want to securely log out of my account so that I can end my learning session and prevent unauthorised access to my account.

Acceptance Criteria:

- Session ends immediately upon logging out
- User is redirected to the login screen
- No personal account data is stored in session storage or cookies after logout

Definition of Done: I am successfully logged out, my session is cleared and I am taken to the login screen.

2.9 Delete Account

User Story: As a learner, I want to permanently delete my account so that my personal details and learning progress are removed from the system.

Acceptance Criteria:

- Option to delete account is clearly visible in account settings
- User is asked for confirmation before deletion
- All personal details and learning progress are removed from the system
- User receives confirmation of account deletion

Definition of Done: My account and all related data are permanently removed and I receive confirmation that the deletion was successful.

2.10 Learning Curriculum

User Story: As a learner, I want to have a structured learning plan so that I can learn more effectively.

Acceptance Criteria:

- There is a clear, well-organised curriculum in place
- Lessons are mapped to specific topics
- Users can view their position in the learning path

Definition of Done: I can view a clear, structured curriculum that guides me from basic to advanced topics.

2.11 Curriculum Lessons by Category

User Story: As a learner, I want to access lessons grouped by category so that I can focus on specific topics systematically.

Acceptance Criteria:

- Lessons are organised under clearly defined categories
- Each lesson covers a specific goal or skill within the category
- Users can track their progress within each lesson
- Categories are locked until the previous category is completed

Definition of Done: I can access and complete lessons one category at a time, unlocking the next only after finishing the previous, to ensure structured and progressive learning.

2.12 Learn Animations

User Story: As a learner, I want to view realistic, interactive sign language animations for each lesson so that I can explore them from different perspectives and better understand how to perform each sign.

Acceptance Criteria:

- Each lesson has realistic, interactive animations demonstrating signs
- Animations can be rotated and zoomed for better learning
- Animations are smooth, high quality, and match lesson content

Definition of Done: I can view and interact with realistic animations in each lesson and they accurately show the sign being taught.

2.13 Quiz at the End of Each Category

User Story: As a learner, I want to take a quiz after completing each category so that I can assess my understanding of what I have learned.

Acceptance Criteria:

- Each category ends with a quiz covering the key concepts
- Quiz questions are relevant and vary in format
- Users receive immediate feedback and scores

Definition of Done: I can complete a quiz after each category to test my knowledge and identify areas for improvement.

2.14 Skill Adaptation

User Story: As a learner, I want the application to adapt to my level of signing experience so that I can learn at a comfortable pace and skip content I already know.

Acceptance Criteria:

- Learners are required to take a placement test to indicate their signing experience
- If the learner is a beginner, the placement test is not required and the learner is placed at the start of the curriculum
- If the learner has prior knowledge, lessons covering already known content are skipped
- After placement, learners must follow the structured curriculum from their assigned level onward without skipping future lessons

Definition of Done: I start learning from a level that suits me, without wasting time on things I already know.

2.15 Learning Progress

User Story: As a learner, I want to view my achievements, day streak and progress so that I can track my learning and stay motivated.

Acceptance Criteria:

- Signs learned, lessons completed, day streak, achievements and overall progress are displayed
- Data is updated after a task is completed

Definition of Done: I am able to view my learning progress.

2.16 Camera Input for Translation

User Story: As a user, I want to input signs using my device's camera so that it can be translated into spoken and written language in real time.

Acceptance Criteria:

- Camera access is granted and working
- Signs are detected and captured

Definition of Done: I have successfully entered a sign via the camera for translation.

2.17 Translate Fingerspelling

User Story: As a user, I want to sign individual letters so that they can be translated into spoken and written language in real time.
Acceptance Criteria:

- System recognises the individual signed letters from the input
- Signs are translated accurately
- Signs are translated in real time
- Translated signs are output as text and audio
- Letters appear in the correct order if multiple are signed

Definition of Done: The letters I have signed are accurately translated and displayed immediately.

2.18 Translate Words

User Story: As a user, I want to sign individual words so that they can be translated into spoken and written language in real time.

Acceptance Criteria:

- System recognises each signed word from the input
- Words are translated accurately
- Words are translated in real time
- Translated words are output as text and audio
- Output matches the intended meaning

Definition of Done: The words I have signed are accurately translated and displayed immediately.

2.19 Translate Phrases & Sentences

User Story: As a user, I want to sign phrases or full sentences so that they can be translated into spoken and written language in real time.
Acceptance Criteria:

- System recognises the signs from the input
- Signs are translated accurately
- Signs are translated in real time
- Translated signs are output as text and audio
- Output matches the intended meaning

Definition of Done: The phrases and sentences I have signed are accurately translated and displayed immediately.

2.20 Nursery Rhymes

User Story: As a learner, I want to access nursery rhymes in ASL so that I can learn sign language in a fun and interactive way using songs that I may already know.

Acceptance Criteria:

- Nursery rhymes are presented with animated signing for each song
- Lyrics are displayed alongside the animations for easy following
- Content is suitable for children and encourages interactive learning
- Users can play and replay the videos as needed

Definition of Done: I can access, watch and interact with ASL nursery rhymes, following the lyrics and animations to learn signs effectively.

2.21 Game

User Story: As a learner, I want to play interactive sign language games so that I can learn and practise signs in a fun and engaging way.

Acceptance Criteria:

- Game is interactive and related to learning and practising sign language
- Game content matches current or previous learning material
- User can start, pause and replay the game
- Game provides scoring to track performance

Definition of Done: I can play a sign language game that is engaging, functional and relevant to my sign language learning progress.

2.22 Cross Device Support (PWA)

User Story: As a user, I want to access the sign language learning app on any device (desktop, tablet, mobile) with a consistent experience so that I can learn anytime and anywhere without issues.

Acceptance Criteria:

- The app is installable as a Progressive Web App on supported devices and browsers
- The app layout and functionality adapt responsively to various screen sizes and orientations
- Loading times and performance are acceptable across devices

Definition of Done: I can install and use the app on desktop, tablet and mobile devices with consistent UI and full functionality without crashes or display issues.

3 Functional Requirements

FR 1: Application must allow user profile creation and customization

FR 1.1: The system must support user sign up with secure and validated account creation

FR 1.1.1: Users must be able to create a personal account with required fields (name, surname, username, email, password) and accept Terms & Conditions

FR 1.1.2: Users must be able to view the full, up-to-date Terms & Conditions at sign up

FR 1.1.3: Duplicate usernames and emails must be prevented during sign up

FR 1.1.4: Passwords must meet security requirements (minimum 8 characters, at least one special character)

FR 1.2: The system must support user login, password recovery, and secure session handling

FR 1.2.1: Users must be able to log in using valid credentials (email and password)

FR 1.2.2: Users must be able to reset forgotten passwords via a secure email link

FR 1.2.3: Users must be able to securely log out, clearing session data and returning to the login screen

FR 1.3: The system must allow users to manage and update their profile

FR 1.3.1: Users must be able to update profile details (name, surname, username, email, password)

FR 1.3.2: Users must be able to upload, change, or remove their profile picture in supported formats (JPG, PNG)

FR 1.3.3: Users must be able to permanently delete their account after confirmation, removing all personal data and learning progress

FR 2: Application must allow users to input visual data

FR 2.1: The system must allow users to provide real-time visual input via device camera with granted permission for translation

FR 3: Application must be able to translate sign language

FR 3.1: Application must support translation at varying linguistic levels, and users must be able to switch between these levels as needed

FR 3.1.1: Application must support translation of fingerspelling

FR 3.1.2: Application must support translation of numbers

FR 3.1.3: Application must support translation of individual words

FR 3.1.4: Application must support translation of sentences

FR 3.2: Translations must be accurate, real-time and output as both text and audio

FR 3.3: Translations must display in the correct letter/word order and convey the same meaning as the input sign(s)

FR 4: Application must provide a structured curriculum

FR 4.1: There must be a clear overview of the categories in the curriculum

FR 4.1.1: The course overview must clearly indicate completed, in-progress and locked categories

FR 4.1.2: There must be thematic categories (e.g., alphabets, numbers, colours) for contextual learning

FR 4.2: Categories must be organised progressively, from basic to advanced topics

FR 4.2.1: Each category must unlock only after prerequisites are completed

FR 4.3: Categories must include key areas such as fingerspelling (letters), vocabulary (words and phrases) and sentence construction

FR 4.4: Each lesson in a category must include clear objectives and interactive content

FR 4.4.1: Each lesson must provide realistic, interactive animations that can be rotated and zoomed

FR 4.4.2: There must be general tips and guidance on how to improve overall signing skills

FR 4.5: Provide a quiz at the end of each category with varied question formats (text and video input) and immediate feedback and scoring

FR 4.5.1: Users must receive immediate feedback and quiz results

FR 5: Application must accommodate learners of all signing experience

FR 5.1: Include a placement test to assess signing experience level

FR 5.1.1: Beginner learners must be introduced to signing at a steady pace

FR 5.1.2: Advanced learners must be able to skip beginner lessons

FR 5.2: Ensure learners follow the curriculum sequentially from assigned level without skipping future lessons

FR 6: Users must see their learning progress

FR 6.1: Application must provide comprehensive progress analysis

FR 6.1.1: The app must show the user's daily streak, lessons completed, signs learned, achievements and overall progress

FR 6.1.2: Progress must be presented in a simple, graphical manner

FR 6.2: Progress must update dynamically after task completion

FR 7: Application must provide a built-in game (Sign Surfers)

FR 7.1: Game content must supplement learning and test sign language knowledge

FR 7.2: Users must be able to start, pause, replay and track performance in the game at any time

FR 7.3: The game must allow players to move left, right and jump to navigate obstacles (cars)

FR 7.4: The game must display the target word that the player is trying to complete

FR 7.4.1: Players must be able to collect ASL fingerspelling signs to form the given words

FR 7.4.2: At random intervals, players must be prompted to input a sign using their device camera as part of completing the word

FR 7.5: Players must have a limit of three lives per run, with the game ending when all lives are lost

FR 7.5.1: A life must be lost if a player fails to avoid a car

FR 7.5.2: A life must be lost if the player collects an incorrect sign

FR 7.5.3: A life must be lost if the player inputs an incorrect sign via the device camera

FR 7.6: Scores must be based on the number of correctly completed words and the distance run

FR 8: Application must provide nursery rhymes in ASL as an interactive way for children to learn sign language

FR 8.1: Songs must be familiar and engaging, using sign animations alongside the lyrics

FR 8.2: Users must be able to play, replay and follow along with the signs

FR 9: Application must provide a Help and Support section

FR 9.1: Users must be able to access a Help page containing guides and FAQs

FR 9.1.1: The Help page must be clearly accessible from any page on the app

FR 9.2: Users must be able to access contact/support options for unresolved issues

FR 10: Application must support cross-device access as a Progressive Web App (PWA)

FR 10.1: The app must be installable as a PWA on desktop, tablet and mobile devices

FR 10.2: The app must adapt layout and functionality responsively to screen sizes and orientations

FR 10.3: Acceptable loading times and performance must be ensured on all supported devices

4 Quality Requirements

Note: Quality requirements are ordered from highest to lowest priority.

4.1 Usability

Usability focuses on how intuitive, efficient and supportive the system is for users across all interactions: helping them learn, translate, navigate, and complete tasks with confidence and minimal confusion.

To ensure the system usability, it will follow these parameters:

1. **System Learning Features**
   - **Requirement:** When a user requires assistance in any page of the application, the system shall provide a clearly visible and accessible help section that can be opened within one click or tap.
   - **Pass Criteria:** At least 95% of users can successfully locate and access the help page with no more than one click or tap within 1 minute during usability testing.
   - **Test Method:** End-to-end testing to verify that the Help icon is visible and accessible on all pages and is clickable or tappable. Additionally, the test measures the number of interactions required to open the Help page as well as the time taken to load it.
2. **System Efficiency**
   - **Requirement:** When a user is performing a translation and navigates to any other page (e.g., Home, Learn, Profile or Help) to perform a different task, the system shall preserve the translation state so that when returning to the Translate page, the user can view their translations without data loss.
   - **Pass Criteria:** At least 98% of users can resume their translation without data loss after navigating away and returning during usability testing.
   - **Test Method:** End-to-end testing to simulate a user performing a translation, navigating to another page and returning to the Translate page. The test verifies that the translation state is preserved without data loss or reset.
3. **Minimising User Errors**
   - **Requirement:** When a user attempts to navigate away from an incomplete lesson or submits invalid input, the system shall provide clear, informative warnings or error messages and require explicit user confirmation before proceeding with any action that could cause data loss.
   - **Pass Criteria:** At least 95% of users must be able to recognise, correctly respond to and regard the warnings or error messages as clear and helpful during usability testing.

- **Test Method:** Combination of unit testing, end-to-end testing and user surveys. Unit testing to verify that warning and error handling logic triggers under appropriate conditions. End-to-end testing to ensure that the correct warning or error messages are actually displayed in the user interface. User surveys to confirm that the messages are clear, informative and helpful.

4. **Adapting System to User Needs**
   - **Requirement:** When a user wants to start their next lesson, the system shall allow them to access it directly from the Home page using no more than one click or tap, while still providing the option to navigate through the Learn page to select lessons manually.
   - **Pass Criteria:** At least 99.9% of users must be able to start their next lesson directly from the Home page within one click or tap during usability testing.
   - **Test Method:** End-to-end testing to ensure that users can start their next lesson directly from the Home page within one click or tap. The test verifies that the navigation leads to the correct lesson and records the number of interactions to get to this page.
5. **Confidence and Satisfaction**
   - **Requirement:** When a user is engaged in a lesson, the system shall provide real-time feedback and update the user's progress and correctness after the lesson is completed.
   - **Pass Criteria:** At least 99.9% of users must receive real-time feedback during lessons and have their progress correctly updated after lesson completion during usability testing.
   - **Test Method:** Combination of end-to-end testing and user surveys. End-to-end testing to confirm that real-time feedback is provided during lessons and that user progress is correctly updated after lesson completion. User surveys to assess subjective confidence and satisfaction levels.

4.2 Performance

Performance will be based on how well resources are controlled and managed.

1. **Real-Time Translation Response:** An average translation request must complete within 500ms.
2. **System Responsiveness Under Load:** When multiple users are accessing the system simultaneously, the system shall maintain normal response times for all user interactions including page loads, translation and lesson navigation.
3. **Memory and Resource Management:** The system shall maintain a Google Lighthouse above 70% which makes use of First Contentful Paint, Largest Contentful Paint Speed index and more.

4.3 Availability

Availability focuses on how well the system detects faults, recovers from them, and prevents them from occurring.

1. **Uptime:** The system has at least 99% uptime.
2. **Monitoring and Alerting:** When a system health indicator deviates from normal parameters, 99% of issues must automatically trigger alerts within 1 minute of issue detection.
3. **Data Backup and Recovery:** When system data needs to be recovered due to failure or corruption, the system shall restore user data with zero data loss for changes completed.

4.4 Security

Security focuses on how well the system detects and resists attacks.

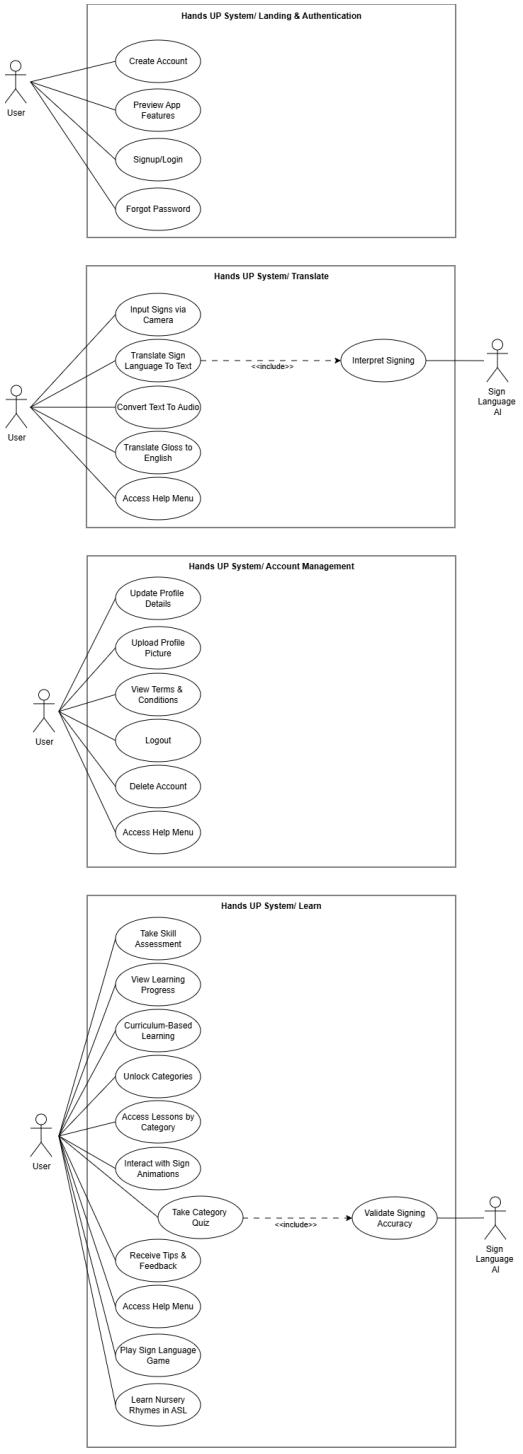
1. **Security Monitoring and Incident Response:** When security threats are detected, the system shall log all security events and trigger appropriate response measures within 5 minutes of detection.
2. **Data Encryption and Communication Security:** When data is transmitted between the client and server, the system shall use HTTPS with TLS 1.3 encryption for all communications including API calls and file uploads.

4.5 Maintainability

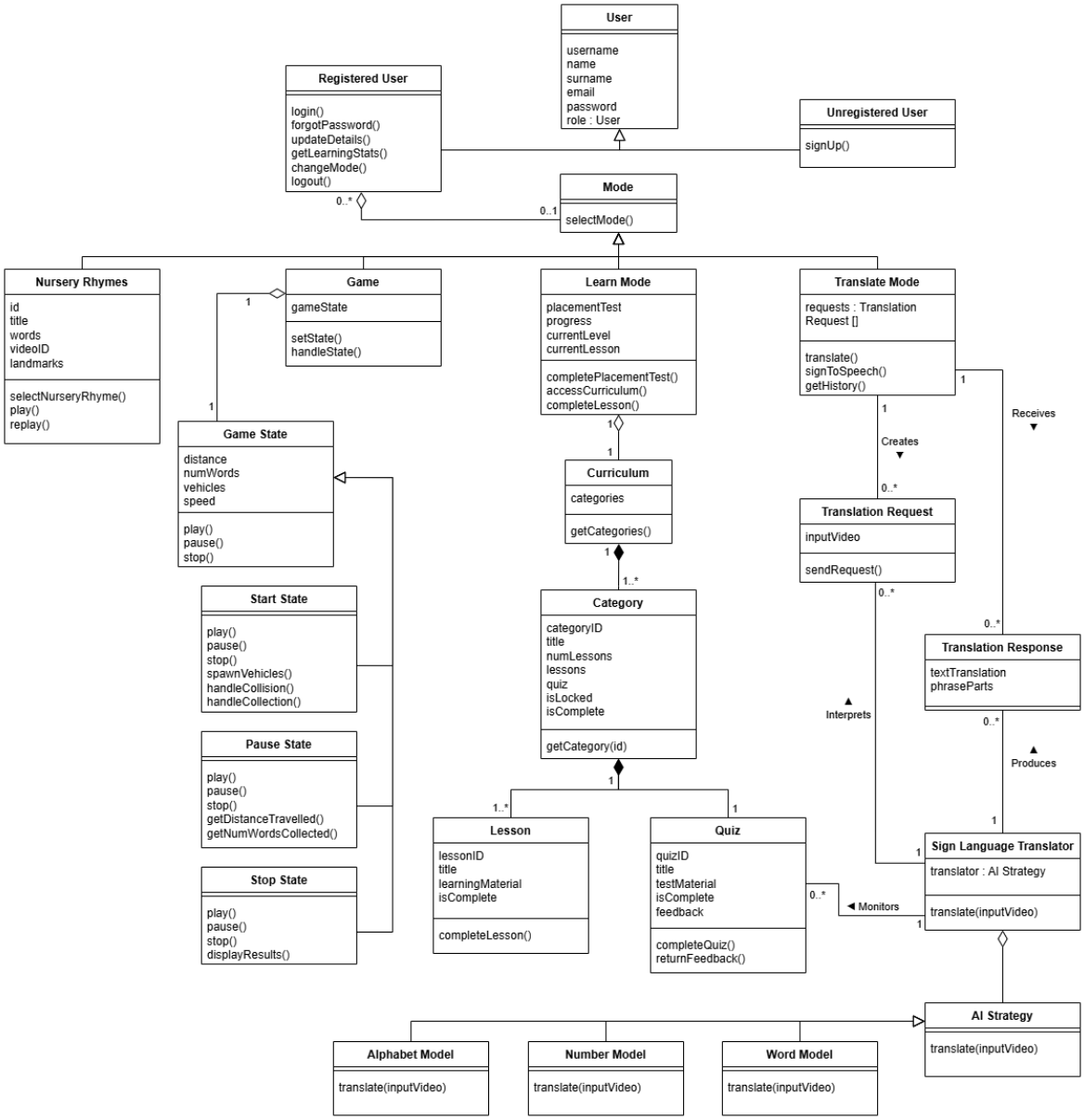
Maintainability focuses on how easily the system can be updated, debugged, and enhanced while maintaining code quality and deployment efficiency.

1. **Code Quality and Coverage:** When new code is added to the system, it shall maintain at least 80% test coverage and pass all static code analysis checks.
2. **Deployment Efficiency:** When deploying system updates, the deployment process shall complete within 10 minutes and include automated rollback capabilities in case of failure.
3. **Bug Resolution:** When bugs are reported, critical issues shall be resolved within 2 days and non-critical ones within 7 days of confirmation.

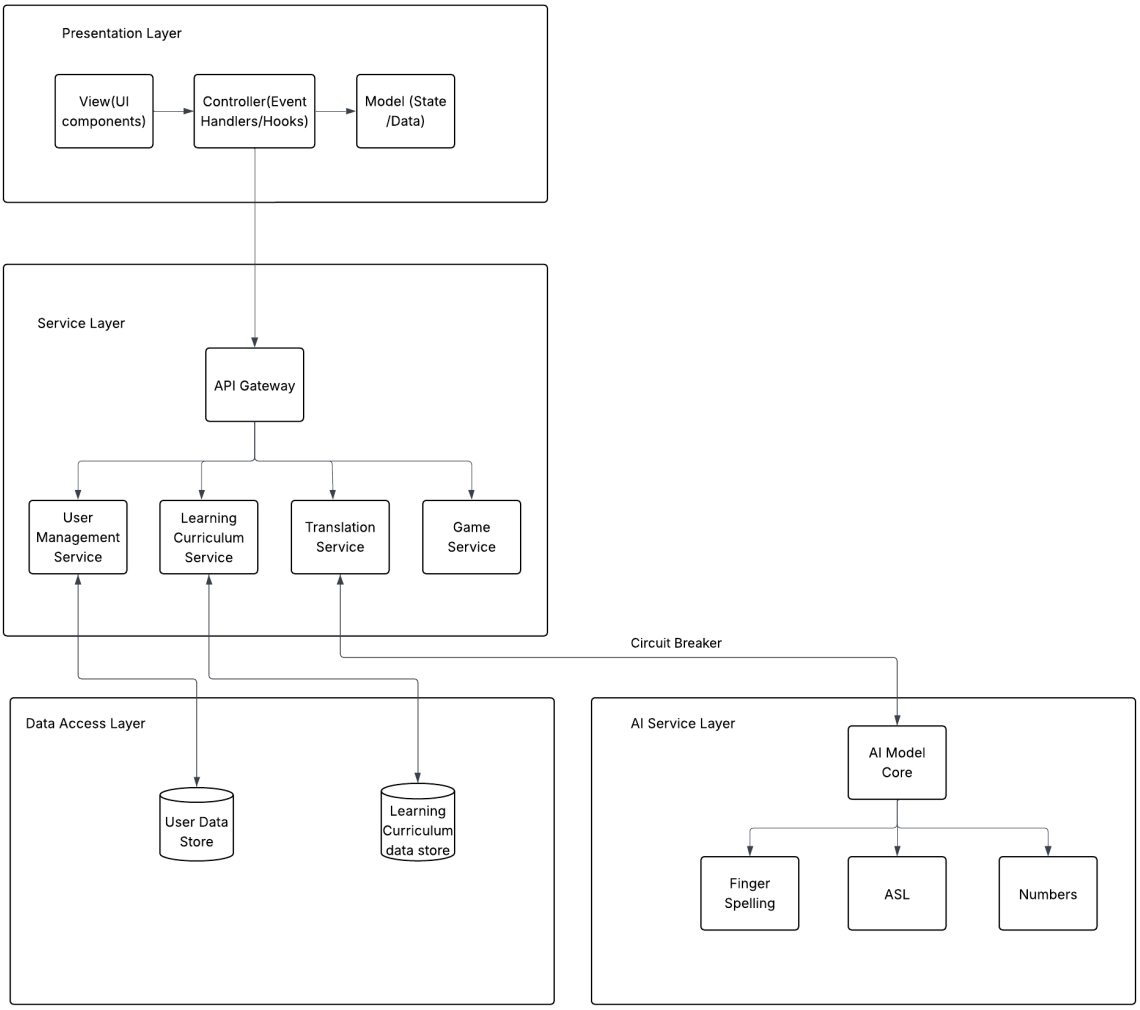
5 Use Case Diagram



6 Domain Model



7 Architecture Diagram



8 Architectural Specification

8.1 Architectural Design Strategy

#	Requirement	Why It Matters
Q1	Usability	The application needs to be intuitive for both new and experienced signers. A smooth, accessible user interface is critical for learning and daily use.
Q2	Performance	The system must deliver real-time sign language translation to be useful. Users expect fast, near-instant responses to their gestures.
Q3	Availability	The translation and learning services must be consistently available. Users rely on the system to learn daily and to bridge communication gaps, downtime would disrupt that.
Q4	Security	Users' data must be protected including personal information and learning progress.
Q5	Maintainability	The system architecture should allow for easy updates and improvements.

Table 1: Quality Requirements and Their Importance

Requirement	Architectural Strategies	Architectural Pattern
Usability	•Responsiveness	Micro-Services
Performance	•Caching •Co-Locate Communicating Resources •Asynchronous Operations	API Gateway
Availability	•Replication	Circuit Breaker
Security	•Detect attacks •Resist attacks •Recover from attacks	API Gateway
Maintainability	•Modularity •Low Coupling •Automated Testing	Microkernel Layered Architecture

Table 2: Architectural Strategies and Patterns

8.2 Architectural Patterns & Justifications

Architectural Pattern	**Justification**
Layered Architecture	Provides clear separation of concerns which ensures security and maintainability. Communication flows sequentially from one layer to the next preventing direct access to the database or core business logic from the user interface.
Service-Oriented Architecture	The backend is structured as a collection of independent, loosely coupled services (e.g. Learning curriculum, User Management and Translation). Each service has a specific responsibility and can be developed, deployed and scaled independently.
Model-View-Controller	It separates the user interface from the application logic and the component handles user input. This separation allows UI updates to be made independently from data logic.
API Gateway	The API Gateway provides a single unified entry point for all client requests. It abstracts the internal service structure, routing requests to the appropriate backend service. It also centralises security controls and ensures requests are handled efficiently.
Circuit Breaker	This pattern prevents a service (translation service) from continuously trying to invoke a dependency that is unresponsive. This prevents the service from making repeated failed requests and allowing it to provide a graceful fallback response to the user.
Microkernel	The AI service is designed with a stable core that handles fundamental AI tasks. Specific sign language models are treated as independent plugins that can be added, updated or replaced without affecting the core AI engine.

Table 3: Architectural Patterns and Justifications

8.3 Architectural Constraints

Client Requirement Constraints

1. **Progressive Web App (PWA):** Must support offline functionality and provide an app-like user experience.
2. **Real-Time Translation:** Response time must not exceed 200ms.
3. **Cross-Device Compatibility:** Accessible via all modern browsers on desktop and mobile devices.
4. **Zero Setup:** No additional installation or configuration required; the app must be usable immediately upon access.

Deployment Constraints

1. **Budget Restriction:** Must use free-tier, open-source or relatively inexpensive solutions.
2. **Cloud-First Approach:** All services must be remotely hosted.
3. **High Availability:** App must maintain at least 99% uptime.
4. **Web-Only Platform:** No native mobile apps; browser-based only.

Technical Constraints

1. **Camera Access:** Exclusively through browser based APIs.
2. **AI/ML Model:** Must be hosted remotely.
3. **Security Compliance:** Must adhere to industry-standard security and privacy regulations.
4. **Performance:** Must support concurrent usage by multiple users without performance degradation.

9 Design Patterns

9.1 State Pattern

Purpose:
The State Pattern is used to manage the different modes and states of the application. Each state is encapsulated as an object that defines its own behavior.

Application:

- **App Modes:** The application supports multiple modes such as *Game Mode*, *Learn Mode* and *Translate Mode*. Each mode behaves differently, and the State Pattern allows smooth switching between them.
- **Game States:** Within the game (Sign Surfers), the State Pattern manages states like *Playing*, *Paused*, and *Game Over*, ensuring consistent transitions and logic.

Benefits:

- Eliminates large conditional blocks.
- Makes it easier to add or modify states and modes.
- Improves code maintainability.

9.2 Strategy Pattern

Purpose:
The Strategy Pattern is used to define a family of algorithms (translation models) and make them interchangeable. This allows the system to switch between models dynamically without altering the core translation logic.

Application:

- The translation feature uses multiple AI models, each specialised for a specific signing complexity:
  - *Alphabet Model* – recognizes and translates individual letters or a series of letters.
  - *Numbers Model* – recognizes and translates numerical signs.
  - *Words Model* – recognizes and translates complete words or sentences.
- The Strategy Pattern allows the app to apply the correct model at runtime.

Benefits:

- Provides flexibility to add new models (e.g., for different dialects) easily.
- Keeps the translation system modular and extensible.

10 Technology Choices

10.1 Database

A relational, SQL database is needed to store structured data such as user details, learning curriculum, lessons, user progress and user results for each lesson. A relational database ensures data consistency through relationships and constraints, making it easier to manage and query complex data.

PostgreSQL	MySQL	Oracle Database
Pros: Open source, can handle complex queries and relationships, strong ACID compliance. **Cons:** Slightly more setup required than MySQL.	**Pros:** Easy to setup and learn, good performance for reading data, well supported on most hosting services. **Cons:** Slightly weaker transaction handling compared to PostgreSQL.	**Pros:** Very powerful and reliable for huge systems, ACID compliant. **Cons:** Expensive and requires licensing, less commonly used with JavaScript.

Table 4: Comparison of SQL Databases

Final Choice: PostgreSQL
Justification: Handles structured and relational data very well, which fits the basic need. It has strong ACID compliance to ensure that all data stays consistent and reliable. Integrates easily with Node.js and Python, making data access and management smooth. It is widely supported by various hosting platforms, which makes deployment, scaling, and maintenance easier in the long run.

10.2 AI Model

The AI model takes in images and videos of sign language and processes them to recognise the signs being shown. It then translates these signs into text. The model should be accurate, efficient and scalable to allow multiple sign languages.

Javascript	Java	Python
Pros: Runs directly in the browser, models can run on any device, useful for real-time AI models. **Cons:** Not suitable for high-performance or complex AI models involving heavy image/video processing.	**Pros:** Good for large scale systems. **Cons:** Limited AI/ML libraries for computer vision.	**Pros:** Best AI/ML support for computer vision and deep learning, offers a wide range of libraries for image/video processing, simplifies model training and testing. **Cons:** Slower runtime.

Table 5: Comparison of Languages for AI Model Implementation

Final Choice: Python
Justification: Offers many libraries and frameworks for training and evaluating models. Works well with TensorFlow, OpenCV and MediaPipe for data preprocessing, training, and visualisation, therefore making it easier to build a custom CNN with high accuracy.

10.3 Frontend

The frontend will be a Progressive Web App (PWA) that allows users to translate sign language to text and learn sign language. The UI should be easy to use, consistent and responsive across different devices.

Plain HTML & CSS	React	Vue.js
Pros: full control over the app's behaviour, lightweight and fast to load. **Cons:** Slower development, harder to maintain and scale.	**Pros:** Good for consistency across devices, provides UI components, supports PWAs, works well with real-time features like camera access. **Cons:** Code can get a bit long and complicated if not organised well.	**Pros:** Easy and fast setup, tools for managing app data, built-in PWA plugin. **Cons:** Does not have as many UI libraries as React, might not be able to handle complex features or multiple users.

Table 6: Comparison of Frontend Frameworks

Final Choice: React
Justification: Makes building a responsive and easy-to-use app across different devices straightforward. It has lots of ready-made UI components, which helps speed up development. Support for PWAs means the app can work offline and be installed easily. Additionally, it handles real-time features like camera access well, which is important for translating sign language live.

10.4 Backend

The backend will handle data processing, model inference and user management. The backend will translate sign language into text using the AI model, manage user profiles and progress, provide a sign language learning curriculum and retrieve translation and learning data. The backend needs to be reliable, scalable, and fast enough to process data in real-time or near real-time, ensuring smooth user interaction.

Node.js	Python (FastAPI)	Java (Spring Boot)
Pros: Works well with React frontend since both use Javascript, can handle many concurrent users and real-time interactions, efficient data handling through database libraries, can call Python AI model for inference. **Cons:** Not ideal for local AI tasks.	**Pros:** Direct integration with Python AI model, high performance and easy API building, can handle multiple simultaneous requests. **Cons:** Scaling to very large user bases may require additional setup.	**Pros:** Very scalable for large systems, strong type safety and reliability, support for structured data handling and user management. **Cons:** Integrating Python AI model requires separate services and increased overhead, more setup required than other two options.

Table 7: Comparison of Backend Frameworks

Final Choice: Node.js
Justification: Makes development and data sharing between frontend and backend easier. Handles many users and real-time interactions smoothly, which fits the need for fast and responsive user experience. Has a strong environment for building APIs and working with PostgreSQL. Since the AI model is built in Python, Node.js can easily integrate with it by calling the FastAPI service when model inference is needed, keeping the system modular and flexible for future updates.

11 Deployment

The application follows a hybrid cloud deployment model, with core services hosted on **Render** and AI models deployed on **Hugging Face Spaces**.

On Render, the React web app (frontend) provides the user interface, while the Node.js backend API handles business logic, data processing and communication between services. A managed PostgreSQL database on Render stores persistent application data and is accessed securely by the backend.

The AI models, developed in Python, are deployed to Hugging Face Spaces, where they expose inference endpoints. The backend communicates with these endpoints over HTTPS, using secure API tokens stored as environment variables.

Updates to the application are managed through a CI/CD pipeline: pushing code to GitHub triggers automatic rebuilds and redeployments on both Render and Hugging Face. All components communicate securely via HTTPS.

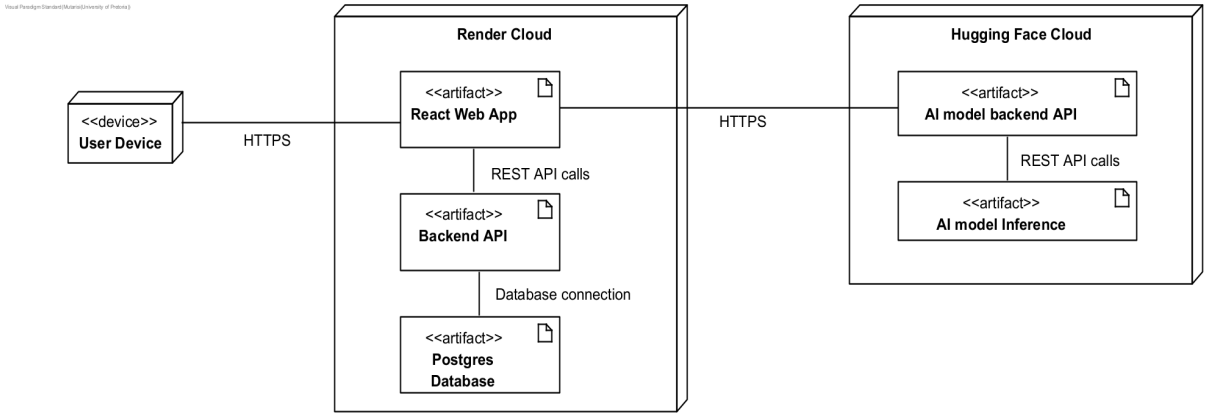


Figure 1: Deployment Model