Avances en el desarrollo de una Aplicación Móvil con Reconocimiento Gestual como Tecnología de Apoyo para la Enseñanza de Cálculo Aritmético en Estudiantes con Necesidades Educativas Especiales

Agustín Álvarez Ferrando[1], Cintia Valero[1], Franco Borsella[1], Juan Etcheverry[1], Matias Batista[1]

[1] Grupo de Investigación & Desarrollo Aplicado a Sistemas informáticos y computacionales
Universidad Tecnológica Nacional, Facultad Regional La Plata

{aaferrando}@frlp.utn.edu.ar
{cintiavalero,fborsella,jetcheverry,mbatista}@alu.frlp.utn.edu.ar

Resumen: En el presente artículo se describen los avances en el diseño y desarrollo de una aplicación móvil, con arquitectura orientada a servicios, la cual se implementará como tecnología de apoyo para la enseñanza de Matemática, promoviendo la inclusión de estudiantes con discapacidad neuromotriz.

El empleo de esta aplicación permitirá a los docentes diseñar situaciones educativas. Entre sus prestaciones más relevantes se encuentran: la personalización de la enseñanza según necesidades específicas de cada estudiante, y la posibilidad de utilizar el reconocimiento facial para interactuar con las actividades a través de gestos.

Durante la etapa inicial del proyecto se refinaron los requerimientos iniciales mediante historias de usuario. Luego, como parte del modelado del sistema se realizaron diagramas de clases y de navegación con UML. Para el proceso de desarrollo, se optó por una arquitectura API REST, y se emplearon tecnologías como React Native para el FrontEnd y Java para el BackEnd, junto con librerías OpenSource para el reconocimiento facial. Se utilizó tecnología Git para la gestión de versiones de código y el servicio GitHub como repositorio.

Finalmente se exponen las conclusiones y se discuten las próximas líneas de acción.

Palabras clave: Accesibilidad, Matemática, Accesibilidad, Microservicios

1. Introducción

En 1978, el informe de Warnock del Reino Unido introdujo el término “Necesidades Educativas Especiales” (NEE), refiriéndose a alumnos que requerían atención educativa diferente para alcanzar su máximo potencial [1]. Las NEE surgen cuando un estudiante tiene un estilo de aprendizaje diferente y necesita recursos adicionales para cumplir con los estándares educativos [2].

La educación inclusiva surge como alternativa a la educación especial. Una escuela inclusiva reconoce y valora las diferencias, minimizando las barreras para la participación de todos los alumnos, independientemente de sus características.

La educación primaria busca recuperar y ampliar los conocimientos matemáticos de los alumnos, estableciendo conexiones entre lo que saben y lo que deben aprender. Los conocimientos numéricos previos de los estudiantes sirven de base para luego avanzar en la comprensión y aplicación de conceptos matemáticos [3].

La discapacidad motora se define como la dificultad para participar en actividades cotidianas debido a problemas para manipular objetos y barreras en el entorno [4]. Para adaptar el currículum a las necesidades de cada niño, se pueden diseñar actividades que utilicen diferentes modalidades sensoriales y recursos tecnológicos, como computadoras o dispositivos móviles que permitan aprovechar funciones avanzadas de accesibilidad, tales como el reconocimiento facial y gestual.

2. Desarrollo e implementación de la aplicación

El proyecto software inició con una etapa de análisis de requerimientos, se delimitaron el alcance y los usuarios principales del sistema (docentes y alumnos).

Para especificar los requerimientos, se utilizó el formato de historia de usuario. Una historia de usuario es una representación de los requisitos de un sistema, escrita en una única frase utilizando el lenguaje natural del usuario. Describen una funcionalidad valiosa para el usuario del sistema [5]. En este caso, se utilizó el

formato COMO *usuario*, QUIERO *funcionalidad*, PARA *beneficio*. En la Tabla 1, se describen algunos de los requerimientos más importantes del sistema, para cada usuario.

Docente	Alumno
COMO docente, QUIERO gestionar unidades temáticas PARA poder cargar actividades relacionadas a cada una de ellas.	COMO alumno, QUIERO poder resolver una actividad PARA ser valorado por el docente.
COMO docente, QUIERO gestionar actividades PARA que los alumnos las resuelvan.	COMO alumno, QUIERO consultar las notas de una actividad PARA conocer el resultado obtenido.
COMO docente, QUIERO revisar las actividades de los alumnos PARA realizar un seguimiento de su aprendizaje.	COMO alumno, QUIERO consultar mi estado académico PARA monitorear mi progreso en el curso.
COMO docente, QUIERO ver el estado de cada alumno PARA realizar un seguimiento de sus avances.	

Tabla 1. Historias de usuario del sistema.

Se utilizaron técnicas de modelado para los diseños del diagrama de clases y diagrama de navegación. Los diagramas de clases permiten describir la estructura del sistema representando los objetos clave en el sistema. A continuación, se ilustran las clases más representativas como **Maestro**, responsable de la administración de los cursos, inscripción de alumnos, creación de actividades y monitorización del progreso de los estudiantes. Y también se identifica el contenido temático y las actividades relacionadas (ver figura 1).

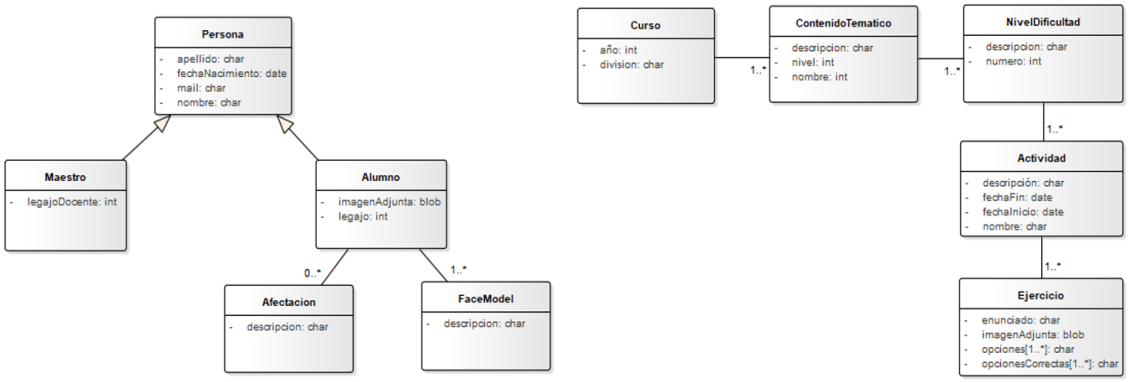

Fig. 1. Diagrama de Clases (Izquierda: Alumno y Docente, Derecha: Actividades)

Para la gestión de versiones [6] de la aplicación se optó por el sistema Git, el cual permite hacer un seguimiento sobre los cambios realizados en el proyecto, y por la plataforma Github, que permite almacenar y manipular repositorios Git. Se trabajó sobre 2 repositorios, uno para el BackEnd y otro para el FrontEnd. Se eligió el flujo de trabajo *Feature Branch* [7], un conjunto de reglas y procesos que utiliza Git para facilitar la colaboración en equipo. Este *workflow* trabaja con ramas que tienen objetivos específicos para facilitar la revisión de código antes que este sea integrado al proyecto.

Para la gestión de tareas se utilizó Kanban, una metodología de trabajo ágil, para aplicar esta metodología se utilizó Trello, una herramienta visual que permite la gestión de tareas y el seguimiento del progreso.

2.1. Arquitectura y tecnologías utilizadas

Para garantizar un sistema interoperable se optó por una arquitectura orientada a microservicios utilizando la tecnología REST [8], para crear interfaces de programación basadas en el protocolo HTTP. REST

establece reglas y restricciones que definen cómo las aplicaciones web pueden comunicarse de manera eficiente y coherente. Se logró así que el producto final escale a dos aplicaciones distintas interconectadas:

- API REST (Lado del Servidor): Facilita la comunicación y transferencia de datos entre el servidor y las aplicaciones cliente. Para su implementación se utilizó lenguaje Java mediante el Framework Spring Boot, aprovechando su capacidad para gestionar solicitudes HTTP de manera eficiente.
- Aplicación Móvil (Lado del Cliente): Ofrece componentes de accesibilidad para que todos los alumnos, independientemente de sus habilidades neuro motrices, puedan participar y resolver ejercicios de manera efectiva. Para su implementación se usó el Framework React Native.

2.2. Diseño de GUI y accesibilidad mediante gestos

Para lograr una interfaz de usuario acorde a los requerimientos del sistema [9] se pensó en un diseño sencillo que garantice la accesibilidad y usabilidad siguiendo los estándares WCAG[1] establecidos por W3C [10]. Algunos de estos puntos incluyen una interfaz intuitiva y simple, con textos cortos y botones grandes que facilitan la interacción; tamaño y espaciado adecuado entre los elementos en pantalla; y opciones de accesibilidad con tecnologías de reconocimiento facial y gestual para el inicio de sesión y la navegación entre pantallas respectivamente.

Se utilizó Figma[2] para el diseño de las pantallas. La figura 2 muestra las pantallas de inicio de sesión con reconocimiento facial, selección de unidad temática y actividad.

Fig. 2. Bocetos de Interfaces de la aplicación hechos en Figma

En cuanto a la accesibilidad y la navegación entre pantallas, se implementó un sistema de reconocimiento facial para facilitar la autenticación de los estudiantes con limitaciones motoras. Para abordar esta situación, se utilizó la biblioteca OpenCV en conjunto con el modelo de Histogramas de Patrones Binarios Locales (LBPH). [11] Se desarrolló un prototipo para la navegación e interacción de los usuarios con la aplicación. Este prototipo incluye la ejecución de tareas mediante gestos. En esta fase inicial, se identificaron tres gestos: el cierre del ojo izquierdo, el cierre del ojo derecho y la una sonrisa. Cada gesto tiene una función específica: el primero se utiliza para retroceder en la aplicación, el segundo para seleccionar el componente enfocado y el tercero para desplazarse entre los distintos componentes de una vista en una dirección determinada. Estas funcionalidades se implementaron utilizando la librería 'expo-face-detector', integrada dentro de la biblioteca Expo.

2.3. Infraestructura y despliegue

Para la gestión de la infraestructura, se emplearon instancias de EC2 de AWS (Amazon Web Services), que son máquinas virtuales escalables en la nube. Se utilizó un entorno para producción y otro para desarrollo.

[1] Iniciativa de Accesibilidad Web cuya función principal es proporcionar recomendaciones para el diseño de páginas web.
[2] Herramienta de diseño gráfico y prototipado web/móvil.

La separación de estos entornos ayuda a realizar pruebas en el ambiente de desarrollo sin afectar el ambiente de producción.

Para integrar y automatizar el proceso de despliegue en estos ambientes, se utilizó Jenkins como la herramienta principal [12]. Jenkins es una herramienta de automatización que se encarga de ejecutar una serie de tareas predefinidas, las cuales varían según el tipo de aplicación que se esté desplegando: BackEnd o FrontEnd.

En el caso del BackEnd, desarrollado en Java con Spring Boot y se administra con Maven, el proceso de despliegue implica la clonación y compilación del proyecto utilizando Maven, una herramienta de gestión de proyectos, seguida de la utilización de Docker para detener el contenedor actual del BackEnd y la clonación del archivo JAR generado por la compilación del proyecto dentro del contenedor. Finalmente, se inicia el contenedor con los cambios realizados en el JAR exponiendo los puertos correspondientes.

Por otro lado, en el FrontEnd, desarrollado en Javascript con React y gestionado con NPM, el proceso de despliegue implica la clonación del proyecto desde el repositorio, la construcción de una imagen Docker para el proyecto, la descarga de todas las dependencias necesarias, seguido de la exposición del puerto correspondiente para el proyecto y la ejecución del script necesario para iniciar el proyecto con Expo utilizando un entorno web. Luego se detiene y elimina el contenedor anterior, y se inicia un nuevo contenedor con los cambios realizados en el proyecto, exponiendo los puertos correspondientes al entorno web.

Además de la infraestructura basada en instancias de EC2 de AWS y la automatización del despliegue con Jenkins, se integró Kong como puerta de enlace, junto con su GUI web, Konga, para facilitar la administración de la infraestructura.

3. Conclusiones y trabajo futuro

En este trabajo se han presentado los avances en el desarrollo de una aplicación móvil accesible que ofrece componentes reconocimiento gestual. Su interfaz sigue criterios del diseño universal y estándares WCAG establecidos por W3C. Los ensayos sobre una primera versión permitieron mejorar el modelo de reconocimiento facial y encontrar algunas limitaciones.

Se espera que a futuro la aplicación permita configurar la acción asociada a cada gesto. Por otro lado, se complementará con una página web para que el docente pueda tener acceso al registro de los alumnos y sus tareas.

Referencias

1. Warnock, M.: *Informe sobre NEE*, Siglo Cero, 130, Madrid, pp. 12-24. (1990)
2. Grijalba Bolaños, J. G., & Estévez Pichs, M. A. *La inclusión escolar un reto para la formación de Licenciados en Educación*. Universidad y Sociedad. (2020).
3. Dirección General de Cultura y Educación de la Provincia de Buenos Aires. "*Diseño curricular para la educación primaria: primer ciclo y segundo ciclo*"; Coord. Sergio Siciliano. 1a ed. La Plata (2018).
4. Pérez, J. I., & Garaigordobil, M. *Discapacidad motriz: autoconcepto, autoestima y síntomas psicopatológicos*. Estudios de psicología, 28(3), 343-357. (2007).
5. Cohn, M. User Stories Applied (1st ed.). Pearson. (2004).
6. Chacon, S., Straub, B. Pro Git v2.1.426. Apress. (2024).
7. Atlassian. Feature branch workflow. Atlassian Git Tutorial. https://www.atlassian.com/git/tutorials/comparing-workflows/feature-branch-workflow. 07/04/2024.
8. Cortés, D. Fundamentos de la arquitectura REST. Medium. https://medium.com/@diego.coder/introducci%C3%B3n-a-las-apis-rest-6b3ad900acc9, 20/02/2024.
9. Pressman, R. Ingeniería del software, Un enfoque práctico 7ma edición. McGraw Hill. (2010).
10. W3C. How To Meet WCAG. https://www.w3.org/WAI/WCAG22/quickref/?versions=2.1#principle1, 25/02/2024.
11. Salton do Prado, K. Face Recognition: Understanding LBPH Algorithm. Medium. https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b, 03/03/2024.
12. Mercedes Brito, J. Integración y entrega continua (CI/CD) con Jenkins. Universidad Oberta de Catalunya. (2022).