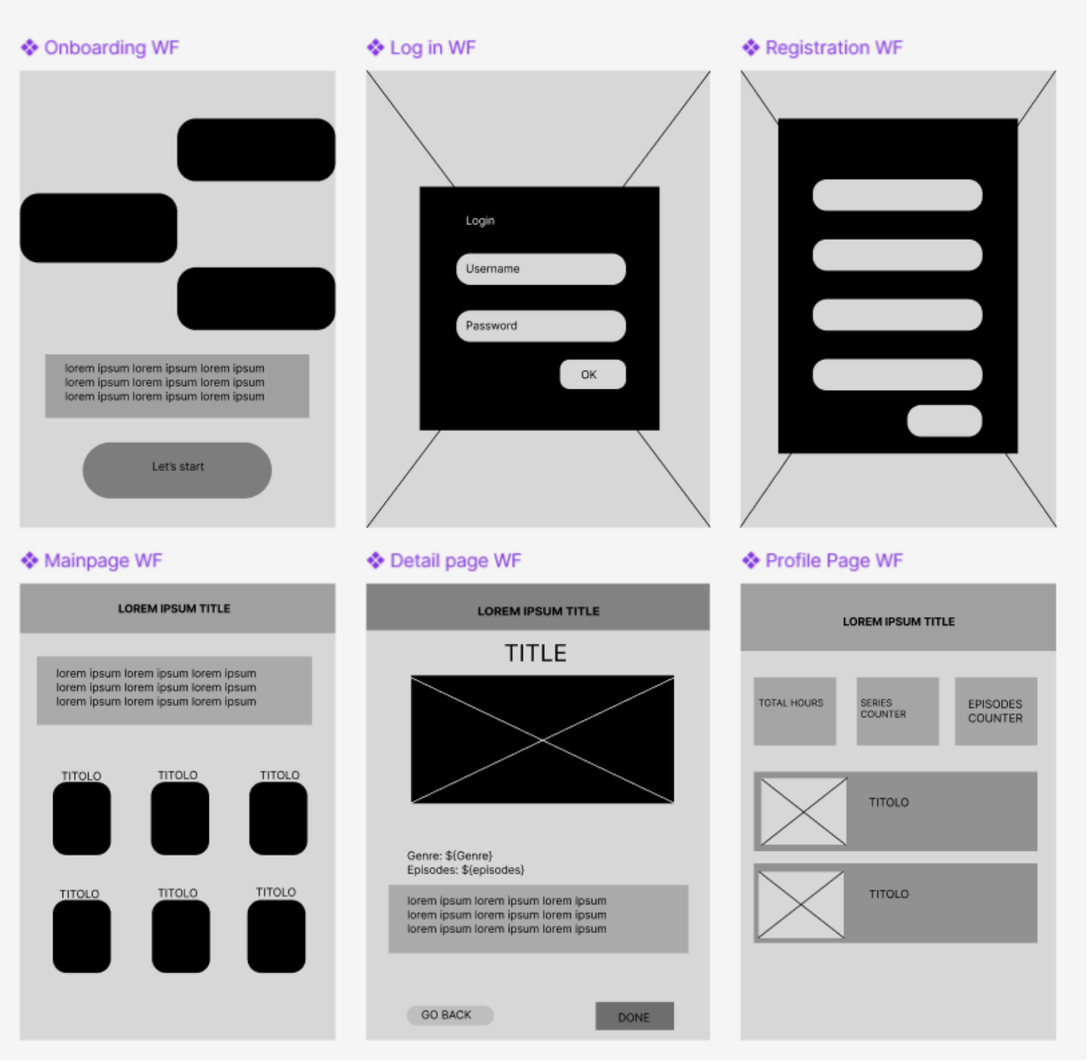
Animedex

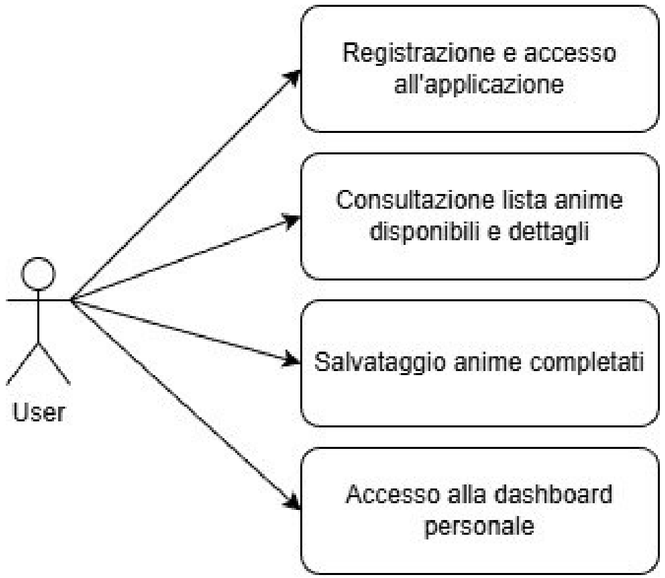
un enciclopedia di anime, per esplorare, consultare tutte le serie animate giapponesi in una applicazione mobile

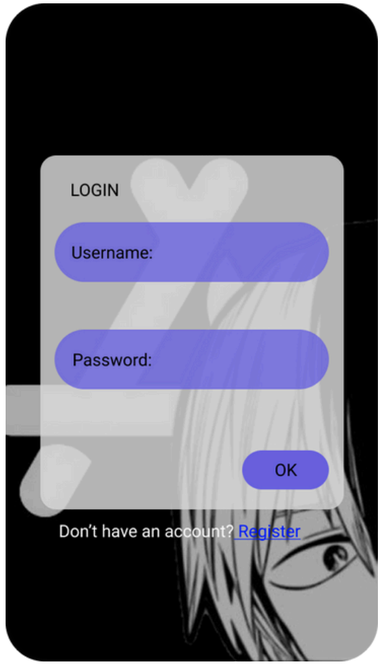
Mark Andro Guevarra

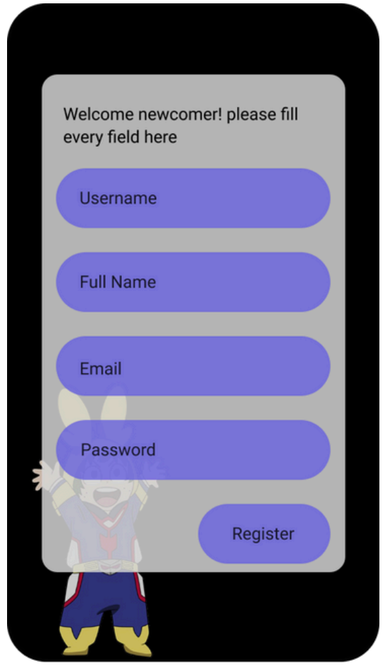
WIREFRAMES

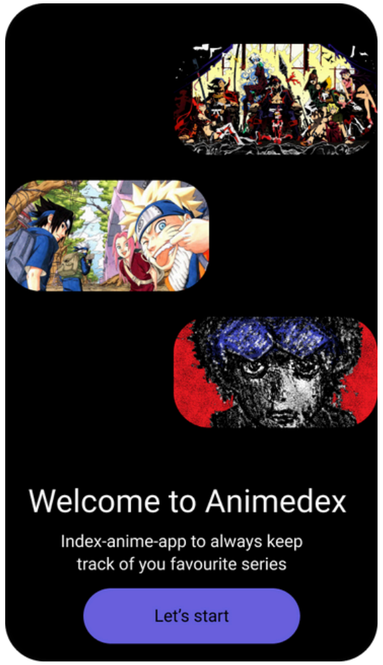

Onboarding WF
lorem ipsum lorem ipsum lorem ipsum
lorem ipsum lorem ipsum lorem ipsum
lorem ipsum lorem ipsum lorem ipsum
Let's start
Log in WF
Login
Username
Password
OK
Registration WF
Mainpage WF
LOREM IPSUM TITLE
lorem ipsum lorem ipsum lorem ipsum
lorem ipsum lorem ipsum lorem ipsum
lorem ipsum lorem ipsum lorem ipsum
TITOLO
TITOLO
TITOLO
TITOLO
TITOLO
TITOLO
Detail page WF
LOREM IPSUM TITLE
TITLE
Genre: ${Genre}
Episodes: ${episodes}
lorem ipsum lorem ipsum lorem ipsum
lorem ipsum lorem ipsum lorem ipsum
lorem ipsum lorem ipsum lorem ipsum
GO BACK
DONE
Profile Page WF
LOREM IPSUM TITLE
TOTAL HOURS
SERIES COUNTER
EPISODES COUNTER
TITOLO
TITOLO

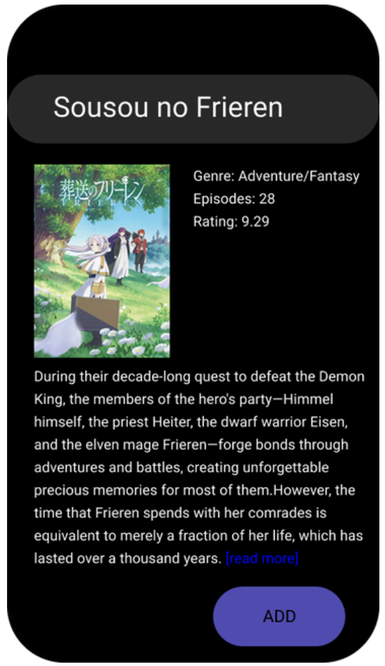
Use cases and design

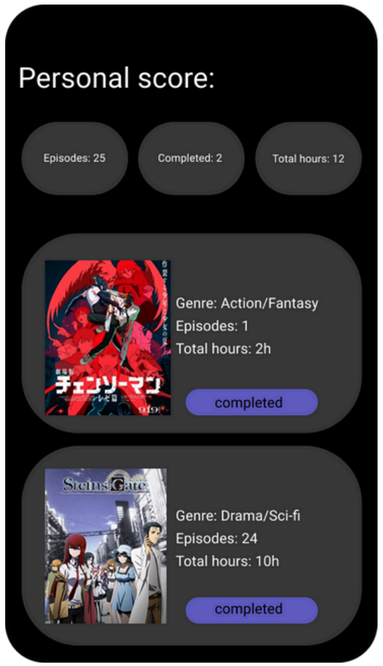












Server

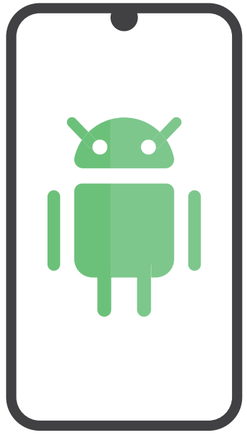
REQUEST

REQUEST

JSON

JSON

Android Studio



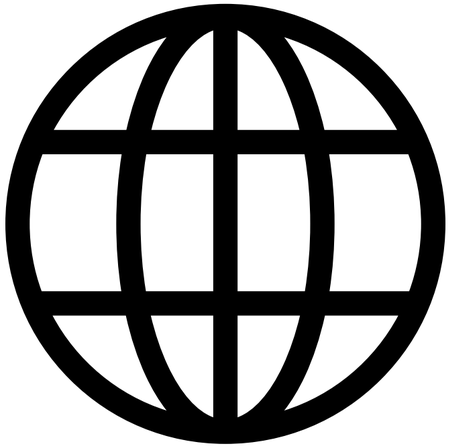
Node.js
Express

ELABORATION

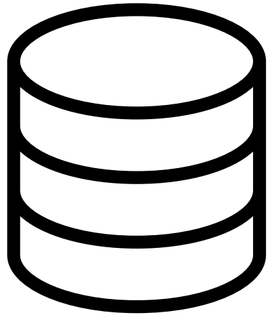
RESPONSE

RESPONSE

Web services



Database



API Jikan

https://api.jikan.moe/v4/anime

Tiramisu - v 13

SQLite

ENDPOINTS

METODO	ENDPOINT	PARAMETRI	RISPOSTA	NOTE	ESEMP
GET	/anime	obbligatori: - opzionali: **page** (default: 1)	JSON con{ success, data:	Restituisce lista anime per punteggio. Sorgente dati https://api.jikan.moe/v4/	GET http://localhost:3000/anime?page=1&limit=25
GET	/anime/search	obbligatori: **q**(parola/e da cercare)	JSON con { success, data:	Se non ci fossero parametri passati ci restituisce errore 400. Sorgente dati	GET http://localhost:3000/anime/search?q=Naruto

GET	/anime/:id	obbligatori:
		id(identificazione di un anime specifico con il suo id)
			JSON con {
			id, title, image, score, synopsis, year, status
			}
				Al click di una risorsa ottiene i dati completi (dettaglio) di un anime tramite Jikan	GET http://localhost:3000/anime/20

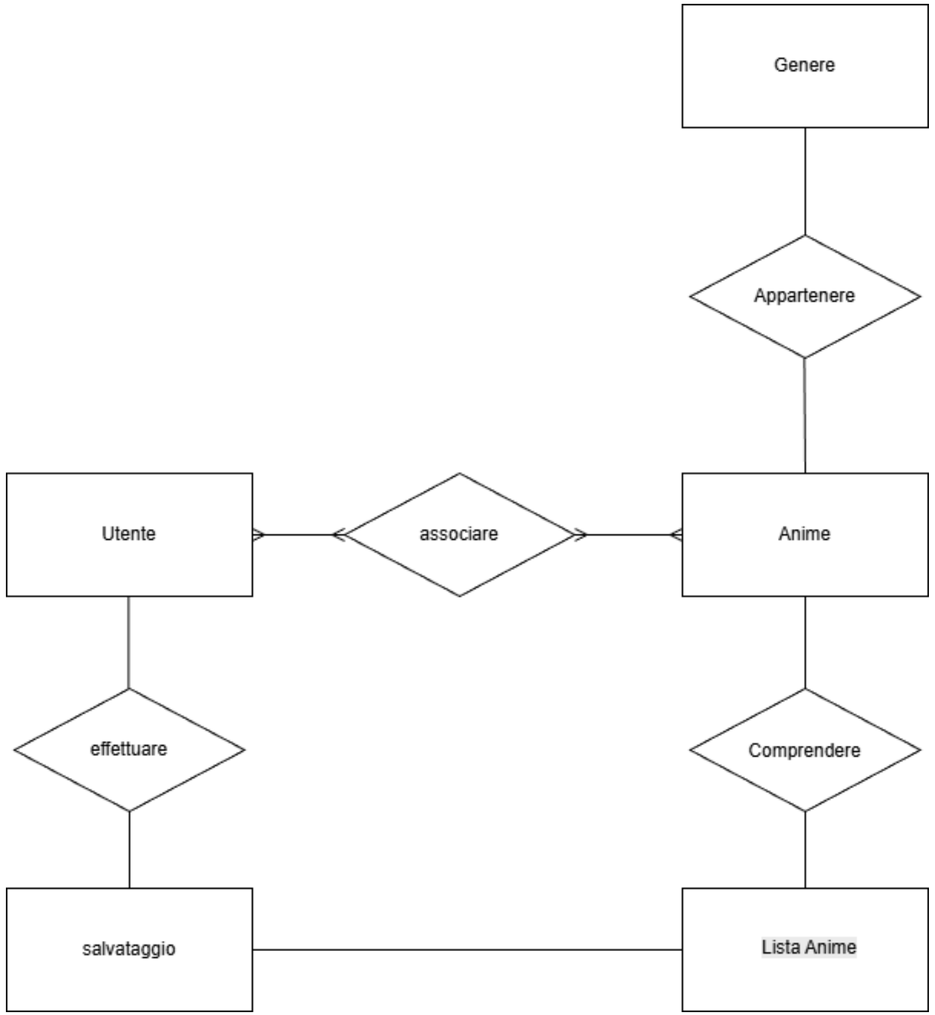
POST	/register	obbligatori:
		nel body JSON
		{“username”: “username”, “password”: “password
			JSON con {
			message, user { id, username}
			}
				Registrazione user	POST
					http://localhost:3000/api/auth/register
					
					BODY:

POST	/login	JSON body: - username (string, obbligatorio) - password (string, obbligatorio)	JSON con {message, user: { id,username} }	Verifica che l'username e la password siano presenti; usa bcrypt per confrontare le password	POST http://localhost:3000/api/auth/login { "username": "mario", "password": "1234" }

POST	/completed	obbligatori:
		nel body JSON
		{“user_id”,”anime_id”,”title” }
			JSON con {
			message, anime{anime_id,user_id,title}
			}
				Salvataggio anime completato nel database	POST
					http://localhost:3000/api/completed

POST	/completed/list	obbligatori:
		nel body JSON
		{“user_id”}
			JSON con {
			message, completed
			}
				Recupero della lista di anime che lo user ha salvato nei completati	POST
					http://localhost:3000/api/completed/list

Modello ER

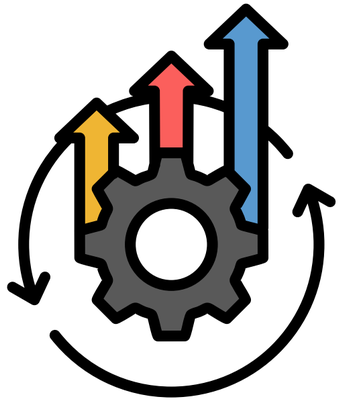


Utente
ID_Utente (PK)
Username
Fullname
Email
Password

Anime
ID_Anime (PK)
Titolo
Descrizione
Genere
Numero_Episodi

Stato Anime
ID_Stato (PK)
ID_Utente (FK)
ID_Anime (FK)
Stato (“Completato”)

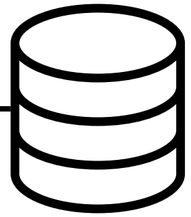
Dashboard Statistiche
ID_Statistica (PK)
ID_Utente (FK)
Totale_Completati



Cambiamenti - implementazione future

Discutere argomenti anime di interesse comune

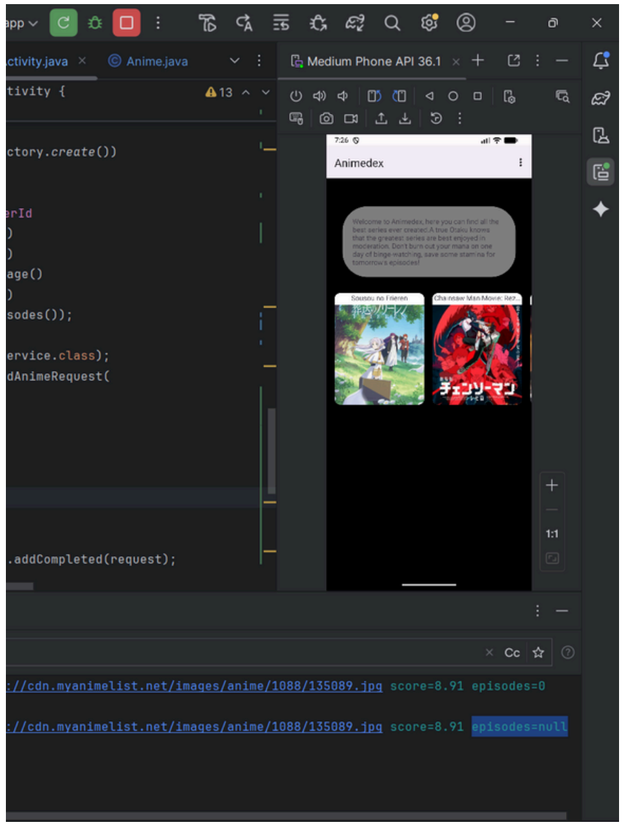
Forum



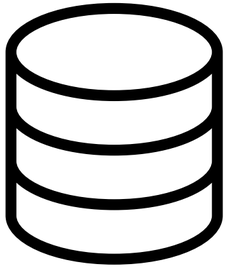
Implementando un database misto

Difficoltà durante la progettazione

1



Aggiunta dati all’interno del database



2

JSON / API SERVICES BACKEND