ELABORATO

INDIRIZZO: ITIA – INFORMATICA E TELECOMUNICAZIONI

ARTICOLAZIONE "INFORMATICA"

ANNO SCOLASTICO 2020-2021

Discipline: INFORMATICA E SISTEMI E RETI

Traccia

Gestione Palestra. Progettare un sistema per la gestione di una Palestra, degli utenti e l'utilizzo delle relative attrezzature.

In particolare, fatte le opportune ipotesi, si chiede di descrivere:

- il progetto dell'infrastruttura tecnologica ed informatica necessaria gestire il sistema, dettagliando:
  - l'architettura di rete e le caratteristiche dei sistemi server;
  - le modalità di comunicazione;
  - gli elementi per garantire la sicurezza del sistema;
- l'architettura dei dati e dei software che si intende implementare, dettagliando il modello dei dati e le modalità di interazione;
- una parte significativa dell'applicazione.

Sommario

ANALISI

Per aiutare le palestre in questo periodo nel quale è necessario il distanziamento sociale, propongo un'applicazione che agevolerà sia gli iscritti in palestra che l'amministratore. In particolare il cosiddetto "amministratore" potrà gestire gli accessi degli utenti in palestra, monitorando in tempo reale il numero di iscritti all'interno e al tempo stesso vedere quali macchinari sono utilizzati. Il sistema in questione mette a disposizione degli utenti, tramite sito web, una pagina login dove, una volta effettuato l'accesso, saranno presenti i propri dati e un **TAG** univoco per ciascuno, il quale tramite la tecnologia **NFC** verrà trasmesso in modalità passiva, che si potrà utilizzare per accedere alla palestra; nell'applicazione è presente anche una pagina dove sono indicati i macchinari, che potranno essere occupati o liberi, a seconda dei rilevatori posti su di essi che captano eventuali segnali dal cellulare. Il rilevatore NFC, una volta che il dispositivo si avvicina, segna data e orario di inizio e data e orario di fine dell'utilizzo, che verranno salvati sul server insieme al macchinario utilizzato. Così l'amministratore potrà anche vedere i macchinari più utilizzati e con più sovraffollamenti. Alla fine della giornata si potrà fare il resoconto degli allenamenti.

MODERN FITNESS

FUNZIONAMENTO DELLA TECNOLOGIA NFC

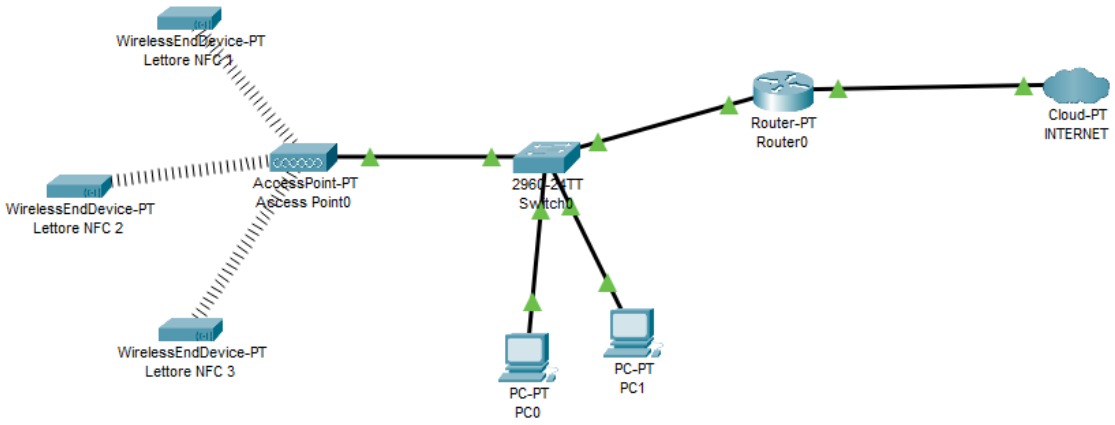
NFC, acronimo di Near Field Communication (comunicazione di prossimità), è una tecnologia che permette di far comunicare tra loro due dispositivi totalmente senza fili semplicemente avvicinandoli tra loro a una distanza di qualche centimetro.

Al giorno d'oggi chiunque abbia un cellulare che sia stato prodotto in questi ultimi anni è dotato di questa nuova tecnologia che permette a un dispositivo di inviare onde ad un ricevitore per **l'identificazione** ed, eventualmente, il tracciamento. Nonostante le dinamiche all'interno di NFC siano molto più evolute, il fine resta più o meno lo stesso: tramite NFC è possibile procedere all'identificazione di due dispositivi abilitandoli al trasferimento di dati semplicemente avvicinandoli tra loro. Questa tecnologia permette il trasferimento di una piccola quantità di dati (424 Kbit per secondo) e solo ad una distanza non superiore ai 3-4 centimetri tra i due dispositivi. Il chip nfc,

all'interno dello smartphone, contiene un **codice univoco** non modificabile che permette di riconoscere l'utente senza possibilità di errore; Essendo il codice non modificabile, non è possibile (o non è sicuramente facile) "clonare" un cip per immedesimarsi in un altro utente;
Nel progetto, questa tecnologia, sarà utilizzata tramite lettori NFC posti sulle macchine; Se un iscritto dovrà utilizzare una macchina basterà appoggiare sopra questo lettore lo smartphone che invierà, tramite protocollo HTTPS, al server una modifica del parametro DISPONIBILE nella tabella ATTREZZO, in questo modo l'attrezzo verrà segnato come occupato. Un altro utilizzo per questa tecnologia è il **controllo degli accessi** all'entrata della palestra, sempre con un lettore NFC, che identificherà l'utente così da poter controllare la data di scadenza del corrispettivo abbonamento.

INFRASTUTTURA DI RETE

L'architettura di questo progetto è di tipo "**three-tier**", ovvero solo il server ha accesso al database, e quindi gli utenti e i rilevatori (client) potranno accedere solo all'interfaccia proposta dal server.



La rete interna alla palestra è suddivisa in 2 **sottoreti**, la prima utilizzata per i dispositivi che si vorranno connettere alla rete della palestra, mentre la seconda per connettere i vari dispositivi NFC utilizzati nella palestra; ogni sottorete avrà la propria **VLAN**, ci sarà la parte dedicata ai PC fissi e quella dedicata ai lettori NFC **wireless**. Per ottimizzare l'utilizzo degli indirizzi IP li suddividiamo in 2 subnet utilizzando parte del campo degli host.

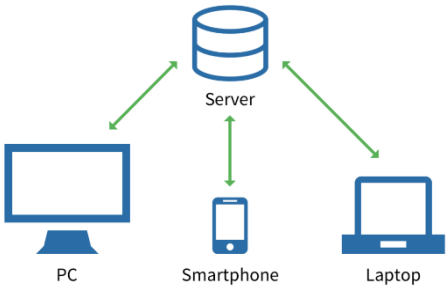
Ho optato per la scelta di lettori NFC wireless così da non avere fili che intralcino nella palestra, questo però porterà alla problematica del posizionamento dell'access-point, perché ci sarà il bisogno di inserirlo in una posizione dove tutti i dispositivi si possano connettere senza perdita di potenza.

Le **trasmissioni wireless** devono contrastare le interferenze che possono essere emesse da altre sorgenti. Le interferenze sono di 2 categorie:

- interferenza da canali adiacenti:
  il segnale emesso è disturbato da un altro di **frequenza** simile; una soluzione tipica è quella di allocare una banda di frequenza per separare le due bande di frequenza adiacenti;
- interferenza da co-canale:
  è dovuta a sistemi vicini che usano la stessa **banda**; una soluzione consiste nell'adottare tecniche di riconoscimento di trasmissione contemporanea sulla stessa frequenza e l'allocazione dinamica del segnale.

Struttura ed utilizzo del server

Il sistema in questione è di tipo distribuito, centralizzato e data-centrico dove c'è il bisogno di mantenere alte ed efficienti le interazioni da parte degli utenti con i dati.



Sul server è necessario quindi:

- Apache, che utilizza il browser per poter interfacciare client e server;
- MySQL, che è utilizzato come database e ha il compito di gestire i dati.

Per quanto riguarda al server c'è la possibilità di scegliere se acquistarlo fisicamente oppure tramite un servizio **cloud**.

- Per quanto riguarda la prima opzione, oltre al costo elevato di un server, potrebbe essere problematico gestire la relativa sicurezza e la possibilità di perdita dati. Essendo un server locale si dovrebbe creare una **DMZ** così da non rendere il server accessibile dall'esterno tramite un servizio specifico del **NAT** chiamato port- forwarding che permette ad un utente esterno di raggiungere un host con indirizzo IP privato, si dovrebbe eseguire il backup dei dati dopo un determinato lasso di tempo, si dovranno implementare uno o più firewall e si potrebbero avere problemi con le capacità fisse del server.

- L'opzione ideale è quindi la seconda, che offre enormi vantaggi come ad esempio la **scalabilità**, che può essere sia verticale che orizzontale (nel primo caso c'è l'utilizzo di una singola macchina alla quale viene aumentata/diminuita la capacità, nel secondo il carico di lavoro è distribuito su diverse macchine) con cui si può modellare il proprio spazio a seconda delle esigenze. La gestione automatica dei backup risparmia alle aziende tempo; il sito è sempre operativo anche in caso di problemi con un server, essendo distribuito su più di un server e la sicurezza dei dati è gestita completamente dal cloud.

Privacy e comunicazione

Un altro aspetto fondamentale da dover considerare è la sicurezza dei dati. Per prima cosa bisogna specificare che i dati personali sono sottoposti alla **privacy**. Il Regolamento Europeo Privacy è una norma obbligatoria che definisce i requisiti minimi che un'organizzazione deve avere per dimostrare la protezione delle persone fisiche con riguardo al trattamento dei dati personali.
Il termine inglese privacy indica la sfera privata degli individui e quindi fa riferimento all'insieme di informazioni personali che non desideriamo diventino di dominio pubblico senza il nostro consenso.
Un dato personale è l'insieme delle informazioni relative ad una persona fisica (cioè al singolo utente) identificata o identificabile tramite diversi tipi di dati:

- I dati identificativi (fotografie, video e qualsiasi cosa permetta l'identificazione diretta dell'interessato);
- i dati anagrafici (nome e cognome, indirizzo mail, indirizzo di residenza e/o domicilio, numero di telefono, ecc.);
- i dati finanziari (codice fiscale, conto corrente, numero carta di credito, ecc.).

La protezione di questi dati è fondamentale, anche dalla possibilità di attacchi esterni. Ci sono molte tipologie di attacco che potrebbero minacciare **l'integrità** e la **persistenza** dei dati, uno tra questi è MITM (man-in-the-middle) in cui qualcuno segretamente ritrasmette o altera la comunicazione tra due parti che credono di comunicare direttamente tra di loro.

Questo problema è stato risolto grazie all'utilizzo dell'HTTPS, al posto dell'HTTP, che si basa sui protocolli SSL e TSL e garantiscono il criptaggio dei dati durante la comunicazione, così da evitare che persone esterne possano stare in ascolto o fare da intermediari; questo sistema di comunicazione viene anche detto end-to-end encryption (EE2E). I protocolli SSL e TSL hanno entrambi una fase iniziale di setup chiamata handshake dove mittente e destinatario decidono quale protocollo di criptografia utilizzare. Finita la fase di setup avviene lo scambio delle chiavi (utilizzando chiavi asimmetriche). Il sistema sarà poi verificato dall'autorità di certificazione che rilascerà il certificato SSL.

METODI PER LA COMUNICAZIONE:

Tramite il protocollo HTTPS utilizziamo i metodi CRUD, con un approccio di tipo RESTful, che verranno utilizzati per poter modificare/reperire le risorse presenti sul server;
Esistono in totale 7 metodi, tra cui ne utilizziamo principalmente 4:

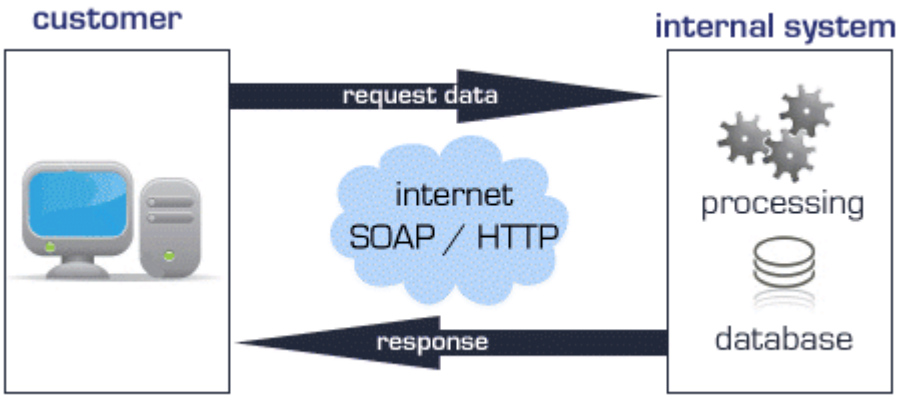
- GET: metodo idempotente e safe, utilizzato per **visualizzare** una rappresentazione di una risorsa;
- POST: metodo non idempotente ed unsafe, utilizzato per la **creazione** di nuove risorse;
- PUT: metodo idempotente ed unsafe, utilizzato per per la **modifica** o l'inizializzazione di una risorsa;
- DELETE: metodo idempotente ed unsafe, utilizzato per **l'eliminazione** di una risorsa.

I metodi aggiuntivi sono:

- HEAD: utilizzato per leggere i metadati di una risorsa;
- OPTIONS: restituisce l'elenco di operazioni possibili;
- PATCH: utilizzato per la modifica di una parte di una risorsa.

I metodi come già detto possono essere idempotenti e safe, la differenza è:

- **Idempotenti** = tutti i metodi che se vengono richiamati generano lo stesso effetto sul server quando vengo applicati da 1 a N volte; Per esempio un metodo idempotente come la GET, se viene eseguita più volte sul server, restituisce sempre lo stesso risultato;
- **Safe** = tutti i metodi che se vengono richiamati non generano alcun effetto collaterale sul server; Per esempio un metodo safe come la GET non applica alcuna modifica sui dati del server, ma restituisce solo una loro rappresentazione grafica.



Sicurezza del database

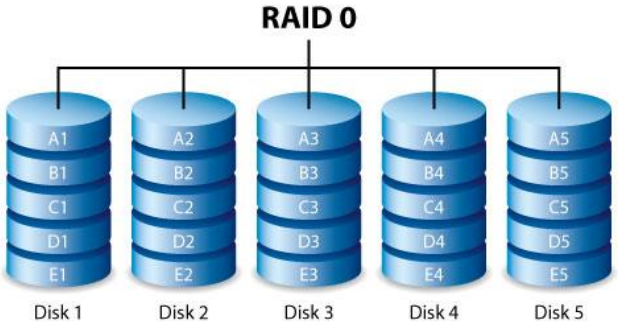
Altre tipologie di attacco molto frequenti sono DoS e DDos, con cui l'attacco informatico fa in modo di sommergere la vittima con pacchetti non chiusi che vanno a sovraccaricare il sistema. Per affrontare questo problema in genere si utilizzano i **firewall**, che seppur non infallibili, sono in grado di filtrare le

richieste, dal momento che analizzano i pacchetti in entrata basandosi su una black-list che contiene gli indirizzi degli host potenzialmente pericolosi (quelli che superano un certo limite di richieste al secondo).
Nonostante i pacchetti provenienti da host malevoli vengano respinti, la loro gestione contribuisce comunque alla congestione del sistema; questo problema si può risolvere inserendo un server **proxy**, che funge da intermediario riesce a smaltire tutte le richieste avendo capacità molto elevate.
Uno dei dati più sensibili sul database sono le **password**. Per la sicurezza di queste non vengono mai salvate in chiaro; il meccanismo di login si basa sul confronto tra la stringa inserita dall'utente e quella salvata nel database, entrambe criptate. Esistono molti metodi per criptare le stringhe, tra cui l'MD5 (uno dei più basilari), di tipo one-way, ovvero la stringa calcolata non può essere decriptata. Uno degli algoritmi più utilizzati e più sicuri al giorno d'oggi è l'AES; lavora su blocchi a dimensione fissa di 128 bit. Ha una chiave di 128 bit, ma possono essere impiegate chiavi più lunghe da 192 e 256 bit per crittare documenti di particolare importanza. La chiave è una stringa alfanumerica che implementa l'algoritmo di codifica/decodifica dell'informazione protetta.
Altre tipologie di attacchi possono essere di tipo **SQL injection** che si verifica quando non viene filtrato l'input dell'utente dai caratteri di escape e quindi vengono passati all'interno di uno statement. Si verifica quando un campo fornito dall'utente non è fortemente tipizzato o non vengono controllati i vincoli sul tipo.

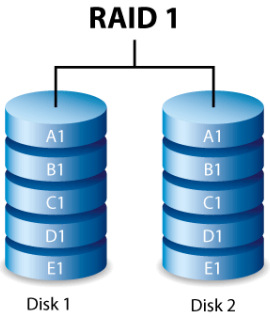
Raid

Il raid è una tecnica di istallazione raggruppata di diversi dischi rigidi sul server. Lo scopo di questa tecnica è di prevenire e minimizzare in danni in caso di guasto di un disco rigido, incrementare le prestazioni di lettura/scrittura del server e l'aumento di memoria disponibile. In un sistema **data-centrico** la salvaguardia dei dati è fondamentale e grazie alle più moderne di queste tecniche si riescono a ripristinare i dati persi a causa di un guasto di un disco.
I dati, una volta scelto il livello del raid, vengono suddivisi in sezioni di lunghezza uguale e disposti su dischi diversi; Grazie a questa suddivisione, in caso il dato da leggere sia più lungo della sezione, l'operazione di lettura può avvenire in parallelo così da migliorane le prestazioni.
Il livello raid è possibile sceglierlo principalmente tra i primi cinque:

RAID 0: Il sistema RAID 0 **divide i dati equamente** tra due o più dischi ma senza mantenere alcuna informazione di parità o ridondanza che aumenti l'affidabilità. Solitamente usato per aumentare le prestazioni di un sistema, o per la comodità di usare un grande numero di piccoli dischi fisici come se fossero un piccolo numero di dischi virtuali.



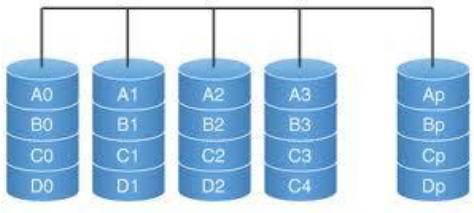
RAID 1: I sistema RAID 1 mantiene una **copia** esatta di **tutti i dati su almeno due dischi**. Solitamente si utilizza quando lo spreco di memoria è meno importante alla salvaguardia dei dati. Lo spazio di memoria viene dimezzato. D'altro canto, visto che un sistema con n dischi è in grado di resistere alla rottura di n - 1 componenti, l'affidabilità aumenta linearmente al numero di dischi presenti.



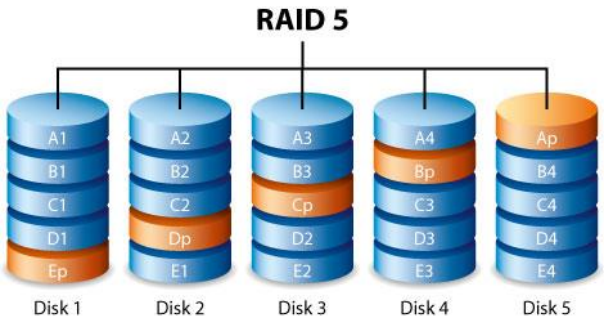
RAID 2: Un sistema RAID 2 divide i dati al livello di bit (invece che di blocco) e usa un codice di **Hamming** per la correzione d'errore che permette di correggere errori su singoli bit e di rilevare errori doppi.

RAID 3: Un sistema RAID 3 usa una divisione al livello di byte con un disco dedicato alla parità. Uno degli effetti collaterali del RAID-3 è che non può eseguire richieste multiple simultaneamente. Questo perché ogni singolo blocco di dati ha la propria definizione diffusa tra tutti i dischi del RAID e risiederà nella stessa posizione, così ogni operazione di I/O richiede di usare tutti i dischi.

RAID 4: Il sistema RAID 4 usa una divisione dei dati a livello di blocchi e mantiene su uno dei dischi i valori di parità dove la suddivisione è a livello di byte. Questo permette ad ogni disco appartenente al sistema di operare in maniera indipendente quando è richiesto un singolo blocco. Il problema principale è quando il disco contenente i valori di parità si guasta.

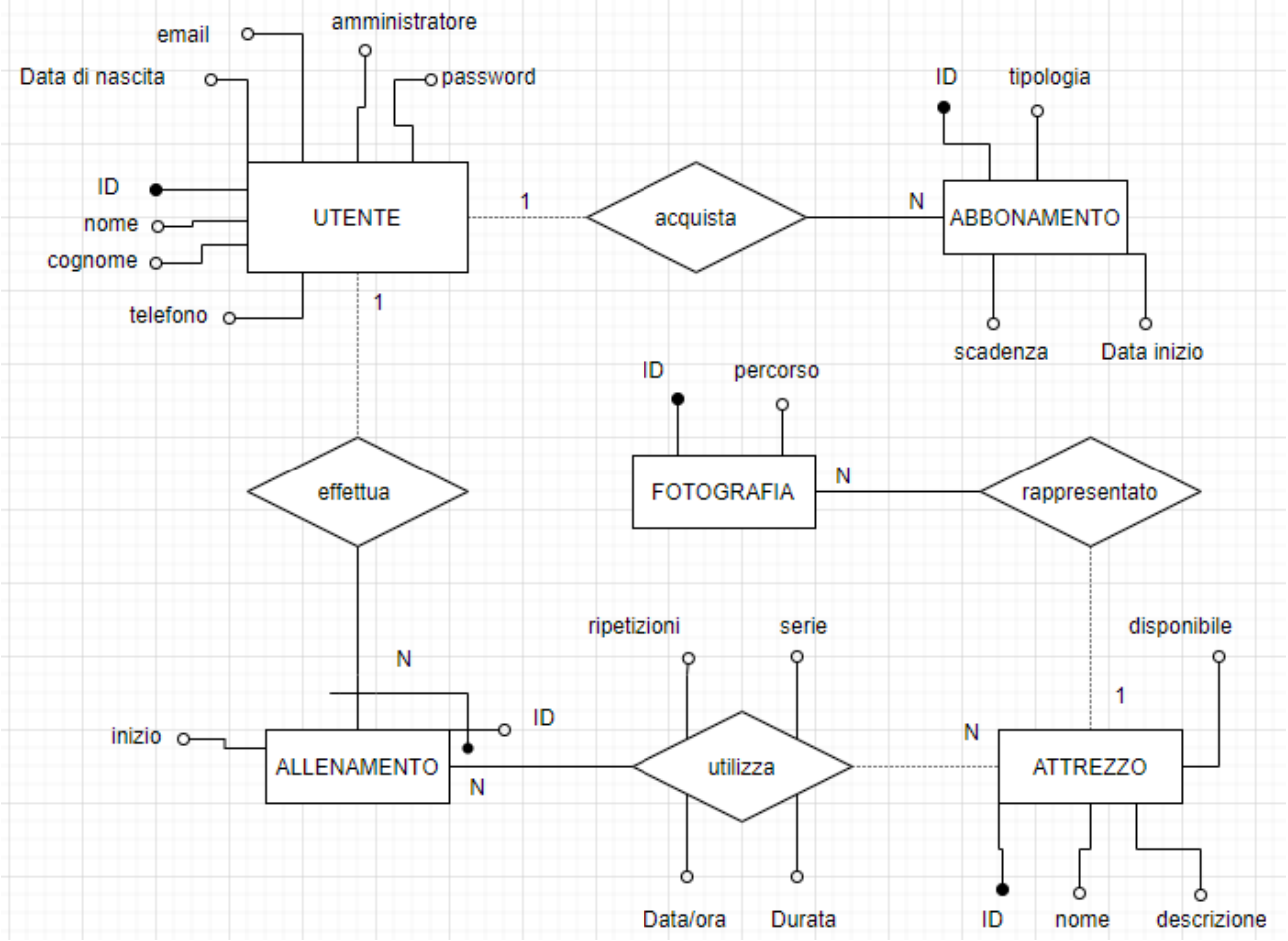


RAID 5: Un sistema RAID 5 usa una suddivisione dei dati a livello di blocco, distribuendo i dati di parità uniformemente tra tutti i dischi che lo compongono. È una delle implementazioni più popolari, sia software, sia hardware. Bisogna notare che il blocco di parità non viene letto quando si leggono i dati da disco, visto che sarebbero un sovraccarico non necessario e diminuirebbe le prestazioni. Il blocco di parità è letto, invece, quando la lettura di un settore dà un errore CRC. In questo caso, il settore nella stessa posizione relativa nei blocchi di dati rimanenti della stripe, insieme al blocco di parità, vengono usati per ricostruire il blocco mancante. In questo modo l'errore di CRC viene nascosto al computer chiamante. Nella stessa maniera, se un disco dovesse danneggiarsi all'interno del sistema, i blocchi di parità dei dischi rimanenti sono combinati matematicamente al volo con i blocchi dati rimasti per ricostruire i dati del disco guasto.



ARCHITETTURA DEI DATI

SCHEMA ER RISTRUTTURATO:



Lo scopo della progettazione concettuale è di costruire e definire una rappresentazione corretta e completa della realtà di interesse. La creazione dello schema ER ci permette la creazione di una rappresentazione astratta e il più possibile formale della realtà.

Osservazioni:

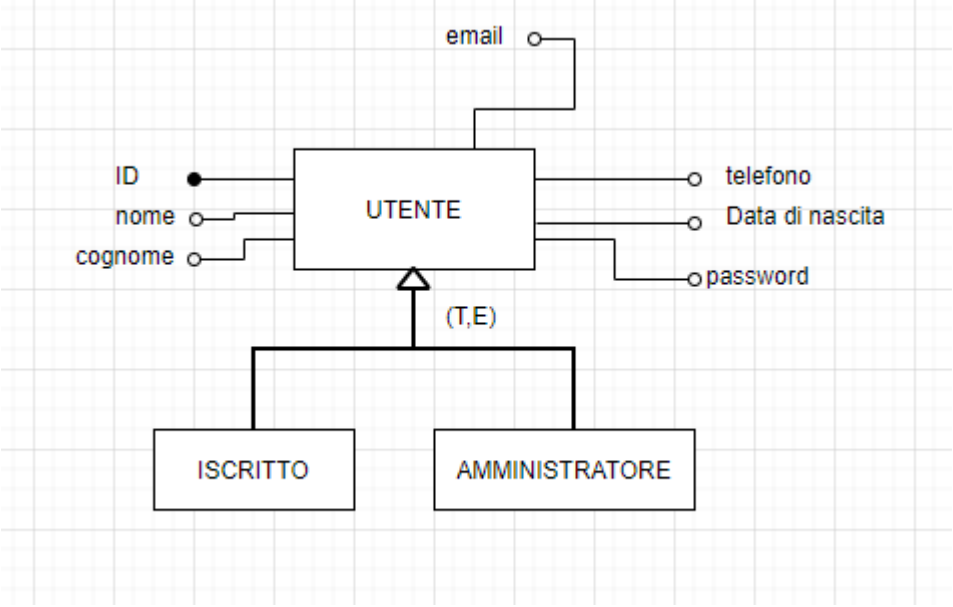
- Un utente, che sia un iscritto o un amministratore, non deve avere un abbonamento. Nel caso dell'amministratore non può avere un abbonamento, mentre un iscritto può scegliere se averlo: in caso che non lo abbia non può effettuare fisicamente l'accesso in palestra (non avendo un TAG d'accesso);
- Un utente può effettuare più allenamenti che si identificheranno tramite **id associato** all'ID dell'utente, in pratica la chiave primaria di allenamento è formata da: (IDUTENTE*,ID);
- Gli allenamenti si potranno effettuare solamente con le macchine, gli esercizi a corpo libero non vengono registrati.
- Viene creata un'ulteriore tabella che si chiama **UTILIZZA** che è l'associazione N a N tra allenamento e attrezzi così che si possano registrare

ogni singolo utilizzo dei macchinari con Data/ora di inizio e durata dell'utilizzo.

- La tabella UTILIZZA ha un ID che identifica ogni allenamento con il singolo attrezzo anche perché ci possono essere più allenamenti con la stessa macchina per ogni singolo utente.
- Una volta effettuato l'accesso in palestra all'utente viene creato un nuovo allenamento, dove si specifica data e ora di entrata in palestra

RISTRUTTURAZIONE:

La ristrutturazione viene applicata sulla tabella utente che teniamo lasciando le 2 entità figlie; essendo una **ISA** totale ed esclusiva potremmo anche tenere senza alcun problema le due entità.



Ho scelto di mantenere la tabella utente togliendo l'amministratore e l'iscritto, aggiungendo semplicemente alla tabella **UTENTE** un attributo amministratore che indentificherà chi avrà l'accesso come semplice iscritto oppure come admin. Ho fatto così per non complicare lo schema ER ed in questo progetto basta una semplice entità. Se un amministratore vuole iscriversi alla palestra deve creare un nuovo utente.

SCHEMA LOGICO:

Lo scopo dello schema logico è quello di trasformare lo schema concettuale (ER) ancora astratto in uno schema logico, ovvero in una rappresentazione efficiente rispetto alle strutture del sistema di gestione che si intende utilizzare.

```
UTENTE (ID, email, password, nome, cognome, telefono,
      dataNascita, amministratore)
ABBONAMENTO (ID, tipologia, scadenza,
      dataInzio, IDUtente*)
ALLENAMENTO (ID, IDUTENTE*, inizio)
ATTREZZO (ID, nome, descrizzione, disponibile)
UTILIZZA (ID, data, durata, ripetizioni, serie,
      IDAtrezzo*, IDAllenamento*, IDUtente*)
FOTOGRAFIA (ID, percorso, IDAtrezzo*)
```

NORMALIZZAZIONE:

Una volta che la base di dati è stata creata ha raggiunto la normalizzazione fino alla **terza forma normale**.
La normalizzazione è un procedimento di tipo graduale che realizza un'ottimizzazione progressiva a partire da relazioni non normalizzate fino a raggiungere un certo livello di
normalizzazione. Per quanto riguarda il raggiungimento delle varie forme normali bisogna verificare che questa base di dati rispetti certi standard:

1. PRIMA FORMA NORMALE: esiste una chiave primaria (un insieme di attributi che identifica in modo univoco ogni t-upla della relazione) ed ogni attributo è definito su un dominio di valori atomico;
2. SECONDA FORMA NORMALE: ogni attributo non chiave dipende funzionalmente e completamente (cioè non parzialmente) dalla chiave primaria;
3. TERZA FORMA NORMALE: la relazione non deve possedere attributi non chiave che dipendano funzionalmente da altri attributi non chiave.

Quindi applico la normalizzazione sul database fino alla terza forma normale per evitare eventuali **anomalie**, che possono essere ridondanze o duplicazioni; La quarta e quinta forma normale, in realtà, raramente vengono utilizzate, in quanto ad un incremento di rigore nell'eliminazione della ridondanza corrisponde un degrado delle prestazioni.

CREATE TABLE (DDL):

Il DBMS mette a disposizione il linguaggio DDL (Data Definition Language), utilizzato per **descrivere le caratteristiche** delle categorie di dati presenti nel database e le corrispondenze esistenti tra di esse.

```sql
--
-- Struttura della tabella `abbonamento`
--

CREATE TABLE `abbonamento` (
  `ID` int(11) NOT NULL,
  `tipologia` varchar(32) DEFAULT NULL,
  `inizio` datetime DEFAULT NULL,
  `scadenza` datetime DEFAULT NULL,
  `IdUtente` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Struttura della tabella `allenamento`
--

CREATE TABLE `allenamento` (
  `ID` int(11) NOT NULL,
  `inizio` datetime DEFAULT NULL,
  `IdUtente` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Struttura della tabella `attrezzo`
--

CREATE TABLE `attrezzo` (
  `ID` int(11) NOT NULL,
  `nome` varchar(32) DEFAULT NULL,
  `descrizione` varchar(32) NOT NULL,
  `disponibile` int(1) DEFAULT 1
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Struttura della tabella `fotografia`
--

CREATE TABLE `fotografia` (
  `ID` int(11) NOT NULL,
  `percorso` varchar(255) DEFAULT NULL,
  `IdAttrezzo` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
-- --------------------------------------------------------

--
-- Struttura della tabella `utente`
--

CREATE TABLE `utente` (
  `Id` int(11) NOT NULL,
  `email` varchar(16) NOT NULL,
  `password` char(32) NOT NULL,
  `nome` varchar(32) DEFAULT NULL,
  `cognome` varchar(32) DEFAULT NULL,
  `dataNascita` datetime NOT NULL,
  `amministratore` int(1) DEFAULT 0
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Struttura della tabella `utilizza`
--

CREATE TABLE `utilizza` (
  `ID` int(11) NOT NULL,
  `data` date DEFAULT NULL,
  `durata` int(16) DEFAULT NULL,
  `ripetizioni` int(8) DEFAULT NULL,
  `serie` int(8) DEFAULT NULL,
  `IdAtrezzo` int(11) DEFAULT NULL,
  `IdAllenamento` int(11) DEFAULT NULL,
  `IdUtente` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

ESEMPI DI QUERY:

a) Il numero di allenamenti svolti da un utente in una settimana:

```
SELECT count(*)
FROM utente inner join allenamento
        on utente.ID = allenamento.IDUtente
WHERE allenamento.inizio BETWEEN $dataInizio AND $dataFine
            AND utente.ID = $IDUtenteInserito
```

b) L'attrezzo usato maggiormente in un determinato periodo fornito come parametro:

```
SELECT attrezzo.nome,count(*)
FROM attrezzo inner join utilizza on attrezzo.ID = utilizza.IdAttrezzo
WHERE utilizza.data BETWEEN $dataInizio AND $dataFine
GROUP BY attrezzo.ID, attrezzo.nome
HAVING count(*) >= all (
                        SELECT count(*)
                                FROM attrezzo inner join utilizza on attrezzo
                                .ID = utilizza.IdAttrezzo
                                WHERE utilizza.data BETWEEN $dataInizio AND $d
                                ataFine
                        GROUP BY attrezzo.ID
                    )
```

c) Gli attrezzi utilizzati da un utente in un giorno fornito come parametro:

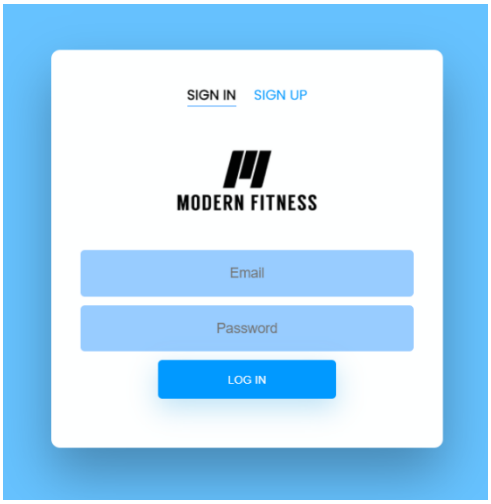
```
SELECT attrezzo.nome, attrezzo.descrizione
FROM attrezzo inner join utilizza on attrezzo.ID = utilizza.IdAttrezzo
WHERE utilizza.IDUtente = $idUtenteInserito AND utilizza.data = $dataInserita
```

d) Tutte gli attrezzi disponibili in questo momento:

```
SELECT *
FROM attrezzo
WHERE disponibile = 1
```

PARTE SIGNIFICATIVA DELL'APPLICAZIONE

Per accedere al sito web bisogna inserire email e password che verranno comparate con i dati sul server. La password inserita dall'utente e viene criptata per poterla comparare con la password criptata presente sul server. Questa interfaccia è raggiungibile tramite **sito web**, quindi può essere accessibile da qualsiasi piattaforma, sia desktop che da smartphone e perciò è necessario che venga implementata tramite un framework responsive, come ad esempio Bootstrap.



I framework responsive sono una tecnica di web design per la realizzazione di siti in grado di adattarsi graficamente in modo automatico al dispositivo coi quali vengono visualizzati.

Dopo aver inserito le credenziali d'accesso c'è bisogno di connettersi con il database, per questo esiste una pagina apposta chiamata connection.php che tramite un oggetto **mysql** apre una connessione verso il database. La connessione al server viene richiesta per poter effettuare le query.

```php
<?php
//file php che crea la connessione al database
$servername = "localhost";
$username = "root";
$password = "";
$dbName = "5a_delogu_ElaboratoPalestra";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbName);
// Check connection
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}
?>
```

Una volta effettuata la login si passa alla pagina dove si visualizzano tutti i macchinari, su questa pagina è presente un filtro che permette all'utente di visualizzare solo i macchinari **disponibili**.

```php
<?php

//query per il filtro
$sql = "SELECT * FROM attrezzo";
//l'attrezzo è disponibile viene visualizzato
```

```
if(isset($_POST["AttrezziDisponibili"]) && $_POST["AttrezziDisponibili"]!="Tu
tti"){

    $sql .= " where disponibile = 1";
  }
$result = $conn->query($sql);
?>
```

Ad ogni utente quando accede in palestra gli viene associato un nuovo allenamento, che poi verrà salvato insieme a tutti gli esercizi nella tabella ALLENAMENTO. La creazione di un nuovo allenamento è gestita dalla pagina chkSession.php.

```
<?php
//pagina php crea per ogni utente che accede in palestra un nuovo allenamento
//parte la sessione
    session_start();
    include("connection.php");
  //se non è settato l'id dell'utente la sessione scade
    if(!isset($_SESSION['userID']))
        header("location: index.php?msg=Sessione scaduta");
//se non è settato il corrispetivo carrello all'utente se ne crea uno
    if(!isset($_SESSION["IDAllenamento"])){
        $sql = "INSERT INTO allenamento (ID, IDUtente, inizio) VALUES
      (NULL, ". $_SESSION["userID"] .", ". $_SESSION["dataInizio"] .")";

        if ($conn->query($sql) === TRUE) {
            $last_id = $conn->insert_id;
            $_SESSION["IDAllenamento"]=$last_id;
            echo "New record created successfully. Last inserted ID is: " .
$last_id;
        } else {
            echo "Error: " . $sql . "<br>" . $conn->error;
        }
    }
?>
```