Progettazione preliminare database clienti Weball

Disclaimer

Questo documento è relativo alla progettazione **preliminare** del nuovo database per il servizio *Weball* sviluppato da Francesco Rocca ed Alessandro Salerno. Questo documento potrebbe essere soggetto a modifiche in base a requisiti tecnici, infrastrutturali, legali o formativi.

Entità e Relazioni

Un'iscrizione viene effettuata da un cliente per un dominio con un certo piano. Un cliente può essere una persona fisica o giuridica.

- Un'iscrizione può avere un solo piano, un piano può essere usato da più iscrizioni (1:N);
- Un'iscrizione appartiene ad un solo cliente, un cliente può avere più iscrizioni (1:N);

Formulazione del database

Per permettere l'esistenza parallela di clienti in forma di persone fisiche e giuridiche, il *super-tipo* Cliente è rappresentato nella propria tabella da un identificatore intero auto-incrementale e da un identificatore del suo status giuridico. I dettagli specifici degli utenti registrati come persone fisiche e come persone giuridiche sono rappresentati in tabelle separate (*sottotipi*) con vincolo di integrità verso la tabella del *super-tipo*.

Si prevede di esternalizzare il servizio di pagamento ed utilizzare la soluzione *Stripe*. L'esternalizzazione potrebbe richiedere modifiche al database in base ai requisiti del fornitore.

Schema logico

Tabella `customers`

Nome	`id_customer`	`c_type`	`email`	`password_hash`	`billing_address`	`registration_date`	`phone`
Tipo	`INT`	`CHAR(1)`	`VARCHAR(255)`	`TEXT`	`TEXT`	`DATE`	`VARCHAR(16)`
Attributi	`AUTO_INCREMENT`	`NOT NULL CHECK (c_type IN ('I', 'O'))`	`UNIQUE NOT NULL`	`NOT NULL`	`NOT NULL`	`NOT NULL`	`UNIQUE NOT NULL`
Chiave	`PRIMARY KEY`						

- `id_customer`: identificatore auto-incrementale intero univoco;
- `c_type`: tipo di cliente: `I` per individuo (persona fisica), `O` per organizzazione (persona giuridica);
- `email`: indirizzo E-Mail del cliente;
- `password_hash`: hash SHA256 della password del cliente;
- `billing_address`: indirizzo di fatturazione del cliente;

- `registration_date`: data di registrazione del cliente;
- `phone`: numero di telefono del cliente. Idealmente utilizzabile per 2FA e per impedire la creazione di account duplicati.

Tabella `organizations`

Nome	`id_customer`	`name`	`hq_address`
Tipo	`INT`	`VARCHAR(255)`	`TEXT`
Attributi	`NOT NULL`	`UNIQUE NOT NULL`	`NOT NULL`
Chiave	`PRIMARY KEY, FOREIGN KEY(customers)`		

- `id_customer`: identificatore intero univoco, agisce come primary key per la tabella locale e come foreign key verso la tabella `customers`;
- `name`: ragione sociale dell'organizzazione;
- `hq_address`: indirizzo della sede dell'organizzazione (non necessariamente uguale all'indirizzo di fatturazione).

Tabella `individuals`

Nome	`id_customer`	`first_name`	`last_name`
Tipo	`INT`	`VARCHAR(64)`	`VARCHAR(64)`
Attributi	`NOT NULL`	`NOT NULL`	`NOT NULL`
Chiave	`PRIMARY KEY, FOREIGN KEY(customers)`		

- `id_customer`: identificatore intero univoco, agisce come primary key per la tabella locale e come foreign key verso la tabella `customers`;
- `first_name`: il nome del cliente;
- `last_name`: il cognome del cliente;

Tabella `plans`

Nome	`id_plan`	`max_hits`	`max_yearly_hits`	`price`	`price_per_req`	`duration`
Tipo	`INT`	`INT`	`INT`	`FLOAT`	`FLOAT`	`INT`
Attributi	`AUTO_INCREMENT`	`NOT NULL`	`NOT NULL`	`NOT NULL`	`NOT NULL`	`NOT NULL`
Chiave	`PRIMARY KEY`					

- `id_plan`: identificatore auto-incrementale intero univoco;
- `max_hits`: numero massimo di richieste che possono essere effettuate dai clienti nell'arco di tempo delimitato dal piano;
- `max_yearly_hits`: numero massimo di richieste effettuabili dai clienti nell'arco di un anno solare;
- `price`: prezzo in Euro del piano da corrispondere all'inizio del periodo;
- `price_per_req`: prezzo in Euro da corrispondere per **ogni richiesta** effettuata dai clienti una volta superato il valore di `max_hits`. Funzionalità applicabile **solo** se non si è superato il limite di `max_yearly_hits`;

- `duration`: durata del piano espressa in mesi (esempio: 1 = il piano si rinnova ogni mese).

Tabella `subscriptions`

Nome	`id_subscription`	`id_plan`	`id_customer`	`domain`	`num_hits`	`activation_date`	`price_ceiling`	`price_due`
Tipo	`INT`	`INT`	`INT`	`VARCHAR(64)`	`INT`	`DATE`	`FLOAT`	`FLOAT`
Attributi	`AUTO_INCREMENT`	`NOT NULL`	`NOT NULL`	`UNIQUE NOT NULL`	`DEFAULT 0`	`NOT NULL`	`DEFAULT 0`	`DEFAULT 0`
Chiave	`PRIMARY KEY`	`FOREIGN KEY(plan)`	`FOREIGN KEY(customers)`					

- `id_subscription`: identificatore auto-incrementale intero univoco;
- `id_plan`: identificatore del piano con vincolo di foreign key verso la tabella `plans`;
- `id_customer`: identificatore del cliente con vincolo di foreign key verso la tabella `customers`;
- `domain`: nome di dominio su cui è valido l'abbonamento;
- `num_hits`: numero di richieste finora effettuate dal cliente nell'ambito del singolo abbonamento (richieste effettuate da dal dominio specificato in `domain`);
- `activation_date`: data di attivazione dell'iscrizione. Utile alla determinazione della scadenza;
- `price_ceiling`: prezzo massimo in Euro che il cliente è disposto a corrispondere per le richieste aggiuntive (una volta superate `plans.max_hits`);
- `price_due`: prezzo in Euro totale che il cliente dovrà corrispondere alla scadenza del periodo previsto dal piano per le richieste aggiuntive effettuate.

Tabella `payments`

Nome	`id_payment`	`id_subscription`	`amount_due`	`date_due`	`status`
Tipo	`INT`	`INT`	`FLOAT`	`DATE`	`CHAR(1)`
Attributi	`AUTO_INCREMENT`	`NOT NULL`	`NOT NULL`	`NOT NULL`	`CHECK (status IN ('C', 'P', 'F'))`
Chiave	`PRIMARY KEY`	`FOREIGN KEY(subscriptions)`			

- `id_payment`: identificatore auto-incrementale intero univoco;
- `id_subscription`: identificatore dell'abbonamento con vincolo di foreign key verso la tabella `subscriptions`;
- `amount_due`: prezzo totale in Euro del pagamento;
- `date_due`: data del pagamento;
- `status`: stato del pagamento. Può essere `C` (Completed - pagamento effettuato), `P` (Pending - pagamento fallito in attesa di riprova automatica) o `F` (Failed - pagamento fallito in attesa di riprova umana).

Ottimizzazione ed utilizzo

Si prevede di utilizzare indici, view, trigger e stored procedures per migliorare le performance e la qualità del database. Si prevede, inoltre, di gestire le race conditions (parallelizzazione) mediante l'utilizzo di transazioni e lock, particolarmente nell'incremento di `subscriptions.num_hits` e nella verifica della di `subscriptions.price_ceiling`.

Ulteriori dettagli verranno aggiunti in eventuali future versioni di questo documento.

Sicurezza e note legali

Si prevede avere l'host di database e web server su una macchina singola in DMZ e di impedire l'accesso al database mediante ACL del Firewall. Si prevede di salvare le password esclusivamente mediante l'hash SHA256 e di utilizzare il protocollo HTTPS per le comunicazioni.
Inoltre, per rispettare i requisiti della normativa GDPR, si prevede di applicare ulteriori accorgimenti che verranno dettagliati in eventuali versioni future di questo documento. Similarmente, si prevede di esternalizzare il salvataggio e la gestione dei dettagli di pagamento presso istituzioni regolamentate ed autorizzate, nello specifico il servizio *Stripe*.