Data requirement analysis

Un’applicazione web consente la gestione di aste online. Gli utenti accedono tramite login e possono vendere e acquistare all’asta. La HOME page contiene due link, uno per accedere alla pagina VENDO e uno per accedere alla pagina ACQUISTO. La pagina VENDO mostra una lista delle aste create dall’utente e non ancora chiuse, una lista delle aste da lui create e chiuse e due form, una per creare un nuovo articolo e una per creare una nuova asta per vendere gli articoli dell’utente. ll primo form inserisce nuovi articoli nel database e il secondo mostra l'elenco degli articoli disponibili nel database e dà la possibilità di selezionarne più di uno. Un articolo ha codice, nome, descrizione, immagine e prezzo. Un’asta comprende uno o più articoli messi in vendita, il prezzo iniziale dell’insieme di articoli, il rialzo minimo di ogni offerta (espresso come un numero intero di euro) e una scadenza (data e ora, es. 19-04-2021 alle 24:00). Il prezzo iniziale dell’asta è ottenuto come somma del prezzo degli articoli compresi nell’offerta. Lo stesso articolo non può essere incluso in aste diverse. Una volta venduto, un articolo non deve essere più disponibile per l’inserimento in ulteriori aste. La lista delle aste è ordinata per data+ora crescente. L’elenco riporta: codice e nome degli articoli compresi nell’asta, offerta massima, tempo mancante (numero di giorni e ore) tra il momento (data ora) del login e la data e ora di chiusura dell’asta. Cliccando su un’asta compare una pagina DETTAGLIO ASTA che riporta per un’asta aperta tutti i dati dell’asta e la lista delle offerte (nome utente, prezzo offerto, data e ora dell’offerta) ordinata per data+ora decrescente. Un bottone CHIUDI permette all’utente di chiudere l’asta se è giunta l’ora della scadenza (si ignori il caso di aste scadute ma non chiuse dall’utente e non ci si occupi della chiusura automatica di aste dopo la scadenza). Se l’asta è chiusa, la pagina riporta tutti i dati dell’asta, il nome dell’aggiudicatario, il prezzo finale e l’indirizzo (fisso) di spedizione dell’utente. La pagina ACQUISTO contiene una form di ricerca per parola chiave. Quando l’acquirente invia una parola chiave la pagina ACQUISTO è aggiornata e mostra un elenco di aste aperte (la cui scadenza è posteriore alla data e ora dell’invio) per cui la parola chiave compare nel nome o nella descrizione di almeno uno degli articoli dell’asta. La lista è ordinata in modo decrescente in base al tempo (numero di giorni e ore) mancante alla chiusura. Cliccando su un’asta aperta compare la pagina OFFERTA che mostra i dati degli articoli, l’elenco delle offerte pervenute in ordine di data+ora decrescente e un campo di input per inserire la propria offerta, che deve essere superiore all’offerta massima corrente di un importo pari almeno al rialzo minimo. Dopo l’invio dell’offerta la pagina OFFERTA mostra l’elenco delle offerte aggiornate. La pagina ACQUISTO contiene anche un elenco delle offerte aggiudicate all’utente con i dati degli articoli e il prezzo finale.

Nota: il design del database non cambia tra versione pure HTML e RIA

Database design(DB ACID compliant)

A diagram of a computer

Description automatically generated

SQL Code

CREATE TABLE IF NOT EXISTS articolo (  
 id INTEGER PRIMARY KEY AUTO\_INCREMENT,  
 nome VARCHAR(200) NOT NULL,  
 descrizione VARCHAR(1000),  
 immagine blob,  
 prezzo INTEGER NOT NULL CHECK (prezzo > 0 AND prezzo < 500000),  
 venditore VARCHAR(100) NOT NULL,  
 FOREIGN KEY (venditore) REFERENCES utente(username),  
 CHECK(*length*(nome) >= 3)  
);  
  
CREATE TABLE IF NOT EXISTS asta (  
 id INTEGER PRIMARY KEY AUTO\_INCREMENT,  
 nome VARCHAR(200) NOT NULL,  
 descrizione VARCHAR(1000),  
 immagine blob,  
 scadenza VARCHAR(50) NOT NULL,  
 rialzo\_minimo INTEGER NOT NULL CHECK (rialzo\_minimo > 0 AND rialzo\_minimo < 100000),  
 venditore VARCHAR(100) NOT NULL,  
 aggiudicatario VARCHAR(100),  
 FOREIGN KEY (venditore) REFERENCES utente(username),  
 FOREIGN KEY (aggiudicatario) REFERENCES utente(username) ON DELETE SET NULL,  
 CHECK(*length*(nome) >= 4),  
 CHECK(*length*(scadenza) >= 4)  
);  
  
CREATE TABLE IF NOT EXISTS asta\_articoli (  
 id\_asta INTEGER NOT NULL,  
 id\_articolo INTEGER NOT NULL,  
 PRIMARY KEY (id\_asta, id\_articolo),  
 FOREIGN KEY (id\_asta) REFERENCES asta(id),  
 FOREIGN KEY (id\_articolo) REFERENCES articolo(id)  
);  
  
CREATE TABLE IF NOT EXISTS offerta (  
 id INTEGER PRIMARY KEY AUTO\_INCREMENT,  
 id\_asta INTEGER NOT NULL,  
 username VARCHAR(100) NOT NULL,  
 prezzo\_offerto INTEGER NOT NULL CHECK (prezzo\_offerto > 0 AND prezzo\_offerto < 1000000),  
 data\_offerta VARCHAR(50) NOT NULL DEFAULT *CURRENT\_TIMESTAMP*,  
 FOREIGN KEY (id\_asta) REFERENCES asta(id),  
 FOREIGN KEY (username) REFERENCES utente(username),  
 CHECK(*length*(data\_offerta) >= 4)  
);  
  
CREATE TABLE IF NOT EXISTS utente (  
 username VARCHAR(100) PRIMARY KEY,  
 pass\_hash VARCHAR(250) NOT NULL,  
 nome VARCHAR(100) NOT NULL,  
 cognome VARCHAR(100) NOT NULL,  
 indirizzo VARCHAR(400) NOT NULL,  
 CHECK(*length*(username) >= 1),  
 CHECK(*length*(pass\_hash) >= 10),  
 CHECK(*length*(nome) >= 2),  
 CHECK(*length*(cognome) >= 2),  
 CHECK(*length*(indirizzo) >= 6)  
);

Application requirements analysis

Pages(views), view components, events(user), actions(server)

Un’applicazione web consente la gestione di aste online. Gli utenti accedono tramite login e possono vendere e acquistare all’asta. La HOME page contiene due link, uno per accedere alla pagina VENDO e uno per accedere alla pagina ACQUISTO. La pagina VENDO mostra una lista delle aste create dall’utente e non ancora chiuse, una lista delle aste da lui create e chiuse e due form, una per creare un nuovo articolo(submit form) e una per creare una nuova asta(submit form) per vendere gli articoli dell’utente. ll primo form inserisce nuovi articoli nel database e il secondo mostra l'elenco degli articoli disponibili nel database e dà la possibilità di selezionarne più di uno(db update). Un articolo ha codice, nome, descrizione, immagine e prezzo. Un’asta comprende uno o più articoli messi in vendita, il prezzo iniziale dell’insieme di articoli, il rialzo minimo di ogni offerta (espresso come un numero intero di euro) e una scadenza (data e ora, es. 19-04-2021 alle 24:00). Il prezzo iniziale dell’asta è ottenuto come somma del prezzo degli articoli compresi nell’offerta. Lo stesso articolo non può essere incluso in aste diverse. Una volta venduto, un articolo non deve essere più disponibile per l’inserimento in ulteriori aste. La lista delle aste è ordinata per data+ora crescente. L’elenco riporta: codice e nome degli articoli compresi nell’asta, offerta massima, tempo mancante (numero di giorni e ore) tra il momento (data ora) del login e la data e ora di chiusura dell’asta. Cliccando su un’asta compare una pagina DETTAGLIO ASTA che riporta per un’asta aperta tutti i dati dell’asta e la lista delle offerte (nome utente, prezzo offerto, data e ora dell’offerta) ordinata per data+ora decrescente. Un bottone CHIUDI permette all’utente di chiudere l’asta se è giunta l’ora della scadenza(click button)(db update) (si ignori il caso di aste scadute ma non chiuse dall’utente e non ci si occupi della chiusura automatica di aste dopo la scadenza). Se l’asta è chiusa, la pagina riporta tutti i dati dell’asta, il nome dell’aggiudicatario, il prezzo finale e l’indirizzo (fisso) di spedizione dell’utente. La pagina ACQUISTO contiene una form di ricerca per parola chiave. Quando l’acquirente invia una parola chiave la pagina ACQUISTO è aggiornata e mostra un elenco di aste aperte (la cui scadenza è posteriore alla data e ora dell’invio) per cui la parola chiave compare nel nome o nella descrizione di almeno uno degli articoli dell’asta. La lista è ordinata in modo decrescente in base al tempo (numero di giorni e ore) mancante alla chiusura. Cliccando su un’asta aperta compare la pagina OFFERTA che mostra i dati degli articoli, l’elenco delle offerte pervenute in ordine di data+ora decrescente e un campo di input per inserire la propria offerta, che deve essere superiore all’offerta massima corrente di un importo pari almeno al rialzo minimo. Dopo l’invio dell’offerta la pagina OFFERTA mostra l’elenco delle offerte aggiornate. La pagina ACQUISTO contiene anche un elenco delle offerte aggiudicate all’utente con i dati degli articoli e il prezzo finale.

JavaScript additional requirements:

-Dopo il login, l’intera applicazione è realizzata con un’unica pagina.

-Se l’utente accede per la prima volta l’applicazione mostra il contenuto della pagina ACQUISTO. Se l’utente ha già usato l’applicazione, questa mostra il contenuto della pagina VENDO se l’ultima azione dell’utente è stata la creazione di un’asta; altrimenti mostra il contenuto della pagina ACQUISTO con l’elenco (eventualmente vuoto) delle aste su cui l’utente ha cliccato in precedenza e che sono ancora aperte.

L’informazione dell’ultima azione compiuta e delle aste visitate è memorizzata a lato client per la durata di un mese.(?)

-Ogni interazione dell’utente è gestita senza ricaricare completamente la pagina, ma produce l’invocazione asincrona del server e l’eventuale modifica solo del contenuto da aggiornare a seguito dell’evento.

Completamento delle specifiche:

Il login va con l’utente e password

Aggiunta di un navbar che da’ all’utente l’opzione di andare da ogni pagina a tutte le altre pagine, e puo’ fare anche logout.

Le pagine che hanno un’asta fanno vedere anche l’autore.

Un utente non loggato puo’ fare ricerca delle aste, ma non puo’ aggiungere un’offerta.

Gli errori e successi vengono visualizzati nella pagina.

HTML Puro:

**Bean**

-articolo

-utente

-Asta

-Articolo

**Controller**

-AcquistoServlet

-DettaglioAstaServlet

-HomeServlet

-LoginServlet

-LogoutServlet

-OffertaServlet

-VendoServlet

**Dao**

-DbManager

**Views**

-articoloCard

-astaCard

-messaggi

-navbar

-acquisto

-dettaglioAsta

-errore

-home

-login

-offerta

-vendo

IFML DIAGRAM

I diagrammi di sequenza sono semplificati: gli errori sono raggruppati tutti in ramo else per clarity, e le frecce di ritorno dal dbManager non sono mostrate.  
  
title Login Process  
 participant User  
 participant Browser  
 participant LoginServlet  
 participant DbManager  
 participant PasswordUtils  
 participant Session  
  
 User->>Browser: Enter login credentials  
 Browser->>LoginServlet: POST /login (username, password)  
 LoginServlet->>DbManager: getUtente(username)  
  
 alt Success  
 DbManager-->>LoginServlet: Return Utente object  
 LoginServlet->>PasswordUtils: verifyPassword(password, passwordHash)  
 alt Password valid  
 PasswordUtils-->>LoginServlet: true  
 LoginServlet->>Session: setAttribute("user", utente)  
 LoginServlet->>Session: setAttribute("isLoggedIn", true)  
 LoginServlet-->>Browser: Redirect to /home  
 Browser-->>User: Display home page  
 else Invalid credentials  
 PasswordUtils-->>LoginServlet: false  
 LoginServlet-->>Browser: Forward to login.jsp with error  
 Browser-->>User: Display error message  
 end  
 else Error or user not found  
 DbManager-->>LoginServlet: SQLException or null  
 LoginServlet-->>Browser: Forward to login.jsp with error  
 Browser-->>User: Display error message  
 end  
  
title Logout Process  
 participant User  
 participant Navbar  
 participant Browser  
 participant LogoutServlet  
 participant Session  
  
 Note over Navbar: Displays "Logout" link when user is logged in  
 User->>Navbar: Click "Logout" link  
 Navbar->>Browser: Request /logout  
 Browser->>LogoutServlet: GET /logout  
 LogoutServlet->>LogoutServlet: request.getSession(false)  
  
 alt Session exists  
 LogoutServlet->>Session: invalidate()  
 Session-->>LogoutServlet: Session terminated  
 else Session doesn't exist  
 Note over LogoutServlet: No action needed  
 end  
  
 LogoutServlet-->>Browser: Redirect to /login  
Browser-->>User: Display login page  
  
title Navigation to vendo page  
 participant User  
 participant JSP  
 participant VendoServlet  
 participant DbManager  
  
 User->>JSP: Click "Vendi un articolo" on home page  
 JSP->>VendoServlet: GET request to /vendo  
 VendoServlet->>VendoServlet: Check if user is logged in  
  
 alt User not logged in  
 VendoServlet-->>User: Redirect to login.jsp  
 else Error  
 VendoServlet-->>JSP: Forward to vendo.jsp with error(s)  
 JSP-->>User: Display error message(s)  
 else Success  
 VendoServlet->>DbManager: getArticoliDisponibiliPerUtente(username)  
 VendoServlet->>DbManager: getAsteUtente(username, false)  
 VendoServlet->>DbManager: getAsteUtente(username, true)  
 VendoServlet-->>JSP: Forward to vendo.jsp with attributes  
 JSP-->>User: Display vendo.jsp with forms and data  
 end  
  
title Create new article flow  
  
 participant User  
 participant JSP  
 participant VendoServlet  
 participant DbManager  
  
 User->>JSP: Submit "Nuovo Articolo" form  
 JSP->>VendoServlet: POST to /vendo with action=createArticolo  
 VendoServlet->>VendoServlet: Extract form parameters  
  
 alt Success  
 VendoServlet->>DbManager: inserisciArticolo(nome, descrizione, imageStream, prezzo, username)  
 VendoServlet-->>User: Redirect to /vendo with success message  
  
 else Error  
 VendoServlet-->>User: Redirect to /vendo with error message(s)  
 end  
  
title Create new auction flow  
 participant User  
 participant JSP  
 participant VendoServlet  
 participant DbManager  
  
 User->>JSP: Submit form  
 JSP->>VendoServlet: POST to /vendo with action=createAsta  
 VendoServlet->>VendoServlet: Extract form parameters  
  
 alt Success  
 loop For each selected article  
 VendoServlet->>DbManager: getPrezzoArticolo(id)  
 end  
 VendoServlet->>VendoServlet: Create Asta object with collected data  
 VendoServlet->>DbManager: inserisciAsta(asta)  
 DbManager-->>VendoServlet: Confirmation  
 VendoServlet-->>User: Redirect to /vendo with success message  
 else Error  
 VendoServlet-->>User: Redirect to /vendo with error message(s)  
 end  
  
title View Auction Details  
 participant User  
 participant JSP  
 participant DettaglioAstaServlet  
 participant DbManager  
  
 User->>JSP: Click on auction link  
 JSP->>DettaglioAstaServlet: GET request to /dettaglioAsta?id=X  
 DettaglioAstaServlet->>DettaglioAstaServlet: Check if user is logged in  
  
 alt User logged in  
 DettaglioAstaServlet->>DbManager: getAstaById(astaId)  
  
  
 alt Auction is closed (has winner)  
 DettaglioAstaServlet->>DbManager: getUtente(asta.getOffertaVincitrice().getUsername())  
  
 end  
  
 DettaglioAstaServlet-->>JSP: Forward to dettaglioAsta.jsp with attributes  
 JSP-->>User: Display auction details page  
 else User not logged in  
 DettaglioAstaServlet-->>User: Redirect to login.jsp  
 else Error  
 DettaglioAstaServlet-->>JSP: Forward to dettaglioAsta.jsp with error(s)  
 JSP-->>User: Display error message(s)  
 end  
  
title Close Auction  
 participant User  
 participant JSP  
 participant DettaglioAstaServlet  
 participant DbManager  
  
 User->>JSP: Click "Chiudi Asta" button  
 JSP->>DettaglioAstaServlet: POST to /dettaglioAsta with id and aggiudicatario  
  
 alt Success  
 DettaglioAstaServlet->>DbManager: chiudiAsta(astaId, aggiudicatario)  
 DettaglioAstaServlet->>DbManager: getAstaById(astaId)  
 DbManager-->>DettaglioAstaServlet: Return updated Asta object  
  
 alt Auction has winner  
 DettaglioAstaServlet->>DbManager: getUtente(asta.getOffertaVincitrice().getUsername())  
 end  
  
 DettaglioAstaServlet-->>User: Redirect to dettaglioAsta?id=X with success message  
 else Error  
 DettaglioAstaServlet-->>User: Redirect to dettaglioAsta?id=X with error message  
 end  
  
title Search and Display Auctions  
 participant User  
 participant JSP  
 participant AcquistoServlet  
 participant DbManager  
  
 User->>JSP: Visit acquisto page or submit search  
 JSP->>AcquistoServlet: GET request to /acquisto?parolaChiave=X  
 AcquistoServlet->>AcquistoServlet: Extract parolaChiave parameter  
  
 alt Success with search keyword  
 AcquistoServlet->>DbManager: getAstePerParolaChiave(parolaChiave, now)  
 AcquistoServlet->>AcquistoServlet: Sort auctions by remaining time  
  
 alt User is logged in  
 AcquistoServlet->>DbManager: getAsteVinteDaUtente(username)  
 end  
  
 AcquistoServlet-->>JSP: Forward to acquisto.jsp with attributes  
 JSP-->>User: Display open auctions and won auctions  
 else No search keyword  
 AcquistoServlet-->>JSP: Forward to acquisto.jsp with empty auctions list  
 JSP-->>User: Display search form only  
 else Error  
 AcquistoServlet-->>JSP: Forward to acquisto.jsp with error message(s)  
 JSP-->>User: Display error message(s)  
 end  
  
title View Auction for Offering  
 participant User  
 participant JSP  
 participant OffertaServlet  
 participant DbManager  
  
 User->>JSP: Click on auction to make offer  
 JSP->>OffertaServlet: GET /offerta?id=X  
  
 alt Success  
 OffertaServlet->>DbManager: getAstaById(astaId)  
 OffertaServlet-->>JSP: Forward to offerta.jsp with asta attribute  
 JSP-->>User: Display auction details with offer form  
 else Invalid auction ID  
 OffertaServlet-->>User: Redirect to acquisto.jsp  
 else Error  
 OffertaServlet-->>JSP: Forward to offerta.jsp with error message(s)  
 JSP-->>User: Display error message(s)  
 end  
  
title Submit Offer  
 participant User  
 participant JSP  
 participant OffertaServlet  
 participant DbManager  
  
 User->>JSP: Submit offer form with price  
 JSP->>OffertaServlet: POST /offerta with astaId and prezzo  
  
 alt User logged in  
 OffertaServlet->>DbManager: getAstaById(astaId)  
  
 OffertaServlet->>OffertaServlet: Validate offer amount  
  
 alt Valid offer amount  
 OffertaServlet->>OffertaServlet: Create new Offerta object  
 OffertaServlet->>OffertaServlet: asta.newOfferta(offerta)  
 OffertaServlet->>DbManager: registraOfferta(astaId, offerta)  
  
 OffertaServlet-->>User: Redirect to /offerta?id=X with success message  
 else Invalid offer amount  
 OffertaServlet-->>User: Redirect to /offerta?id=X with error message  
 end  
 else User not logged in  
 OffertaServlet-->>User: Redirect to login.jsp  
 else Error (invalid astaId or database error)  
 OffertaServlet-->>User: Redirect to "acquista" or /offerta?id=X with error  
 end