Esercizio 1: aste online

Versione HTML pura

- 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 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 una form per creare un nuovo articolo e una nuova asta per venderlo.

L'asta comprende

- l'articolo da mettere in vendita (codice, nome, descrizione, immagine),
- prezzo iniziale
- rialzo minimo di ogni offerta
- e una scadenza (data e ora, es 19-04-2021 alle 23:00).

La lista delle aste è ordinata per data+ora crescente e riporta:

- codice e nome dell'articolo,
- offerta massima,
- tempo mancante (numero di giorni e ore) tra il momento del login e la data e ora di chiusura dell'asta.

Cliccando su un'asta compare

- una pagina DETTAGLIO ASTA che riporta

1.per un'asta aperta 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).

- Se l'asta è chiusa, la pagina riporta

1. i dati dell'asta,
2. il nome dell'aggiudicatario,
3. il prezzo finale e

4.l'indirizzo di spedizione.

La pagina ACQUISTO contiene una form di ricerca per parola chiave.

Quando l'acquirente invia una parola chiave

- la pagina ACQUISTO si ricarica e mostra un elenco di aste aperte (la cui scadenza è posteriore all'ora dell'invio) il cui articolo contiene la parola chiave nel nome o nella descrizione.
- La lista è ordinata in modo decrescente in base al tempo (numero di giorni. ore e minuti) mancante alla chiusura.

Cliccando su un'asta aperta compare la pagina OFFERTA che mostra

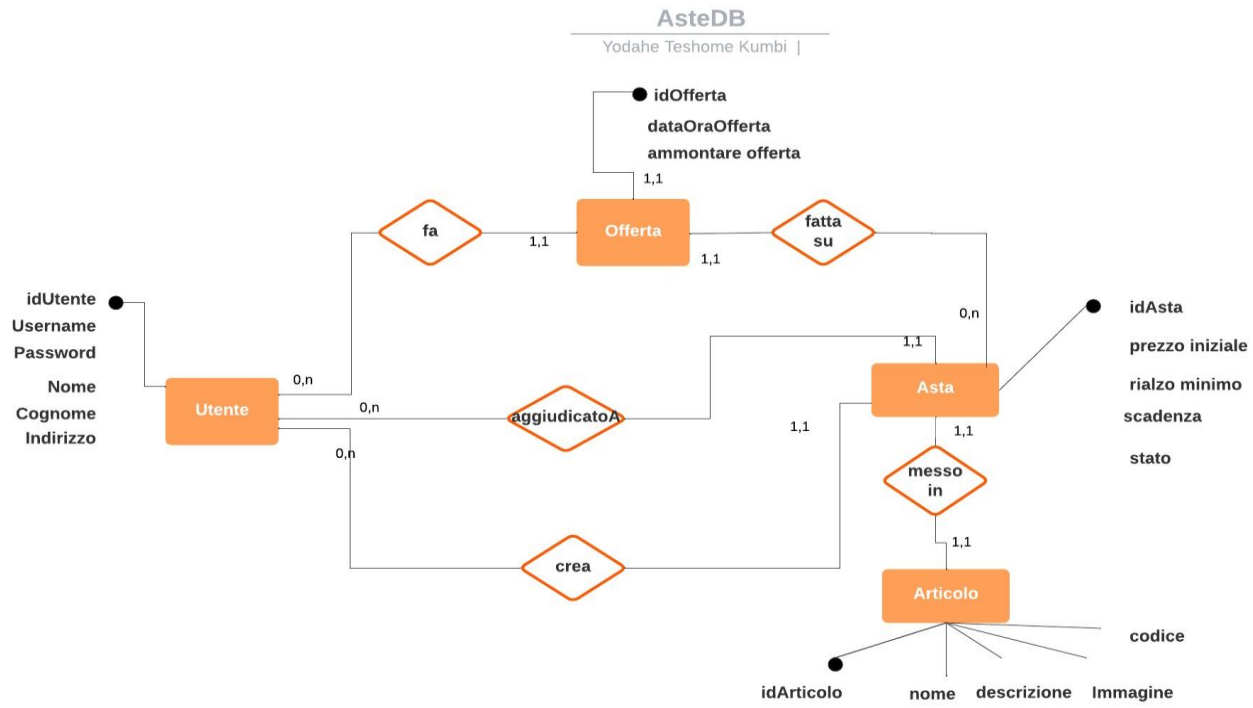
- i dati dell'articolo,
- l'elenco delle offerte pervenute in ordine di data+ora decrescente
- e un campo di input per inserire la propria offerta, che deve essere superiore all'offerta massima corrente.

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 dell'articolo e il prezzo finale

Entities , attributes,relationships

Database design



user (<u>userId</u>, Nome, Cognome, username, password,indirizzo**)**

asta (<u>idAsta</u>,prezzoIniziale,rialzoMinimo,scadenza,status,nome,descrizione,immagine, codice,prezzoAggiudica, aggiudicatoA,**userId)**

offerta (<u>idOfferta</u>, ammontareOfferta, dataOraOfferta**,<u>userId, idAsta</u>)**

Local database schema

```
CREATE TABLE `user` (
  `UserId` int NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  `surname` varchar(45) NOT NULL,
  `username` varchar(45) NOT NULL,
  `password` varchar(45) NOT NULL,
  `indirizzo` varchar(45) NOT NULL,
  PRIMARY KEY (`UserId`)
)
```

```
CREATE TABLE `asta` (
  `idAsta` int NOT NULL AUTO_INCREMENT,
  `prezzoIniziale` float NOT NULL,
  `rialzoMinimo` float NOT NULL,
  `scadenza` datetime NOT NULL,
  `sellerId` int NOT NULL,
  `status` int NOT NULL,
  `prezzoAggiudica` float NOT NULL,
  `vincitoreAsta` int NOT NULL,
  `nome` varchar(45) NOT NULL,
  `descrizione` varchar(45) NOT NULL,
  `codice` varchar(45) NOT NULL,
  `image` longblob,
  PRIMARY KEY (`idAsta`),
  KEY `userid_idy` (`sellerId`),
  KEY `userid_idxy` (`sellerId`),
  KEY `iduser_idx` (`vincitoreAsta`),
  CONSTRAINT `iduser` FOREIGN KEY (`sellerId`) REFERENCES `user` (`UserId`) ON DELETE
CASCADE ON UPDATE CASCADE,
  CONSTRAINT `idVincitore` FOREIGN KEY (`vincitoreAsta`) REFERENCES `user` (`UserId`) ON DELETE
CASCADE ON UPDATE CASCADE)
```

```
CREATE TABLE `offerta` (
  `idOfferta` int NOT NULL AUTO_INCREMENT,
  `ammontareOfferta` int NOT NULL,
  `dataOraOfferta` timestamp NOT NULL,
  `idOfferente` int NOT NULL,
  `idastaFK` int NOT NULL,
  PRIMARY KEY (`idOfferta`),
  KEY `userId_idx` (`idOfferente`),
  CONSTRAINT `userId` FOREIGN KEY (`idOfferente`) REFERENCES `user` (`UserId`) ON
DELETE CASCADE ON UPDATE CASCADE
)
```

Application requirements 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 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 una form per creare un nuovo articolo e una nuova asta per venderlo.

L'asta comprende

- l'articolo da mettere in vendita (codice, nome, descrizione, immagine),
- prezzo iniziale
- rialzo minimo di ogni offerta e una scadenza (data e ora, es 19-04-2021 alle 23:00). La lista delle aste è ordinata per data+ora crescente e riporta:
- codice e nome dell'articolo,
- offerta massima,
- tempo mancante (numero di giorni e ore) tra il momento del login e la data e ora di chiusura dell'asta.

Cliccando su un'asta compare

- una pagina DETTAGLIO ASTA che riporta

1.per un'asta aperta

1. i dati dell'asta
2. 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).

2.Se l'asta è chiusa, la pagina riporta

1. i dati dell'asta,
2. il nome dell'aggiudicatario,
3. il prezzo finale e

4.l'indirizzo di spedizione.

La pagina ACQUISTO contiene una form di ricerca per parola chiave.

Quando l'acquirente invia una parola chiave

- la pagina ACQUISTO si ricarica e mostra un elenco di aste aperte (la cui scadenza è posteriore all'ora dell'invio) il cui articolo contiene la parola chiave nel nome o nella descrizione.
- La lista è ordinata in modo decrescente in base al tempo (numero di giorni. ore e minuti) mancante alla chiusura.

Cliccando su un'asta aperta compare la pagina OFFERTA che mostra

- i dati dell'articolo,
- 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.

Dopo l'invio dell'offerta la pagina OFFERTA mostra -

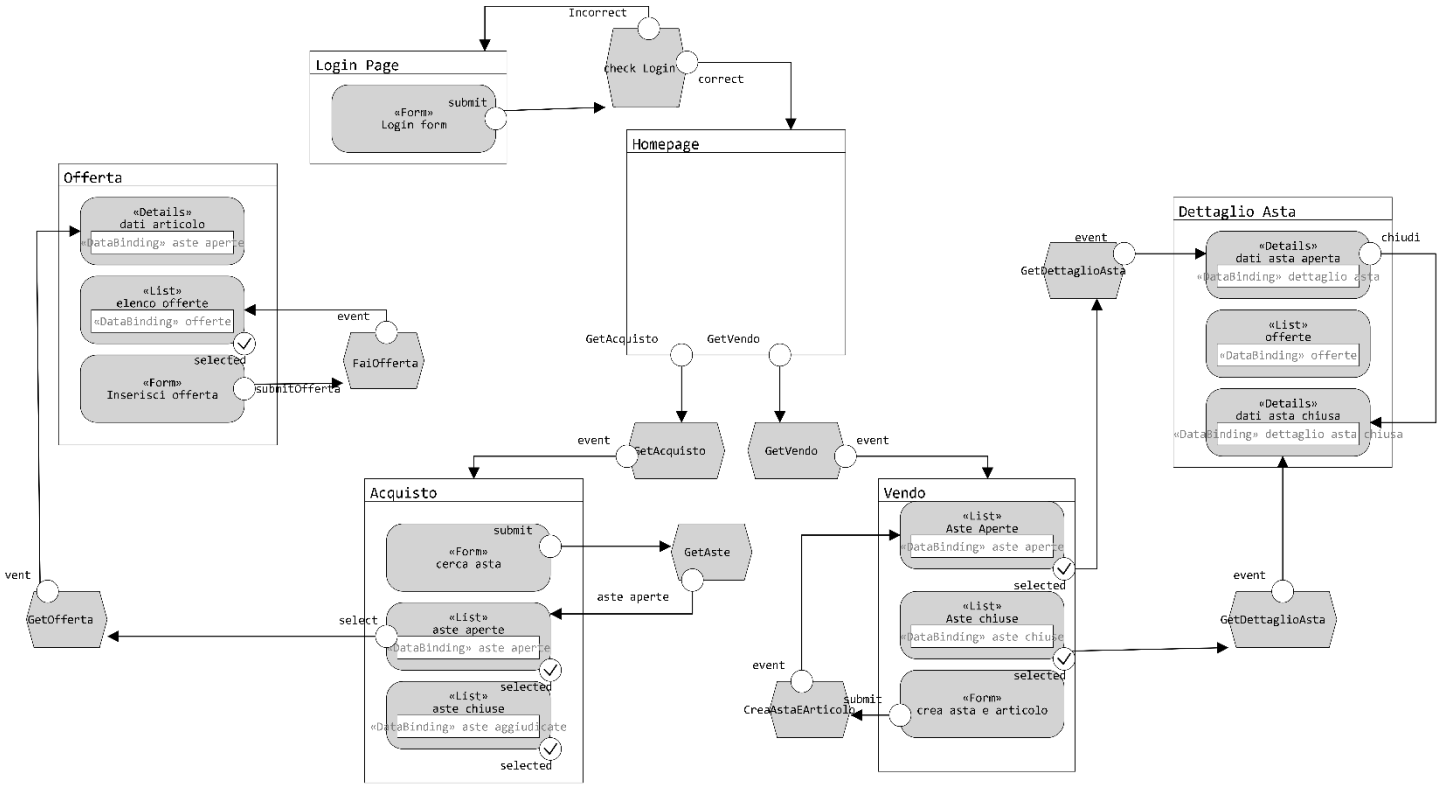
l'elenco delle offerte aggiornate.

La pagina ACQUISTO contiene anche un elenco delle offerte aggiudicate all'utente con -

i dati dell'articolo

- e il prezzo finale
- Pages (views), view components, events, actions

Application design

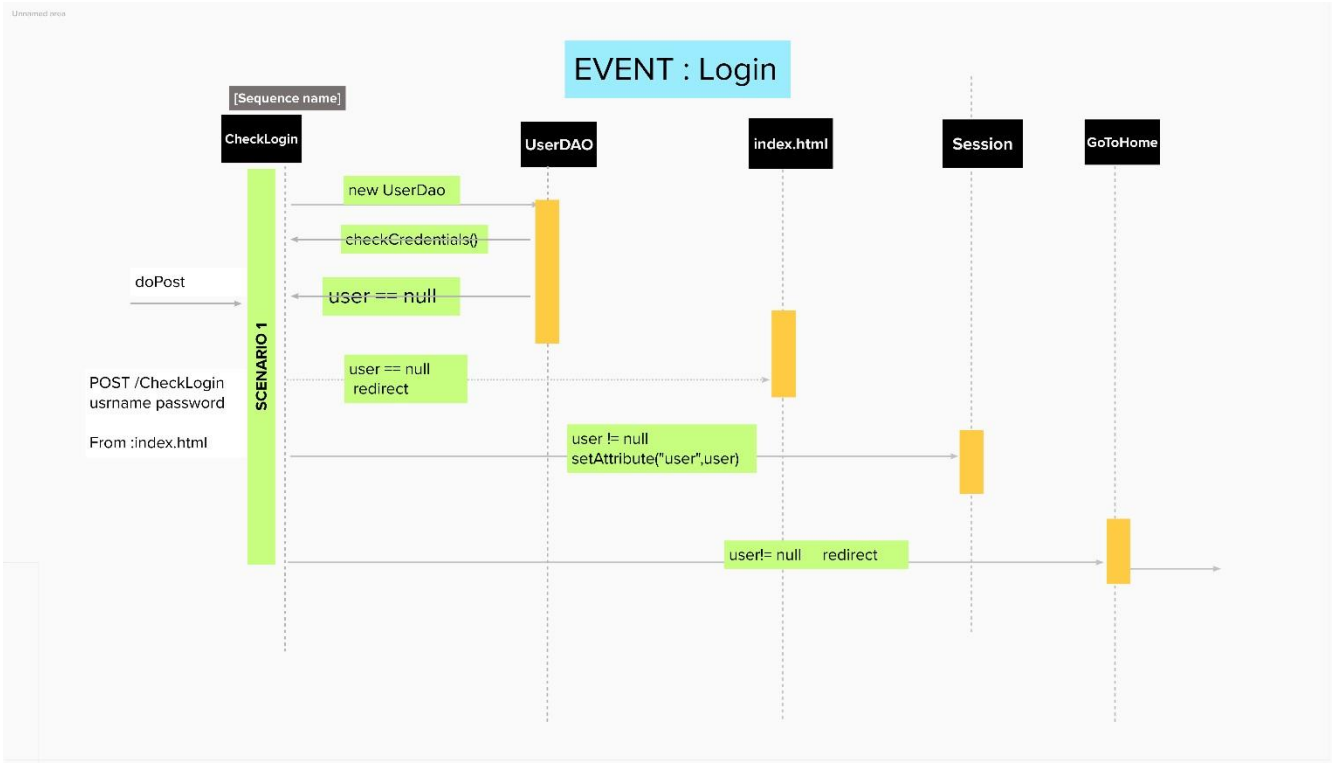


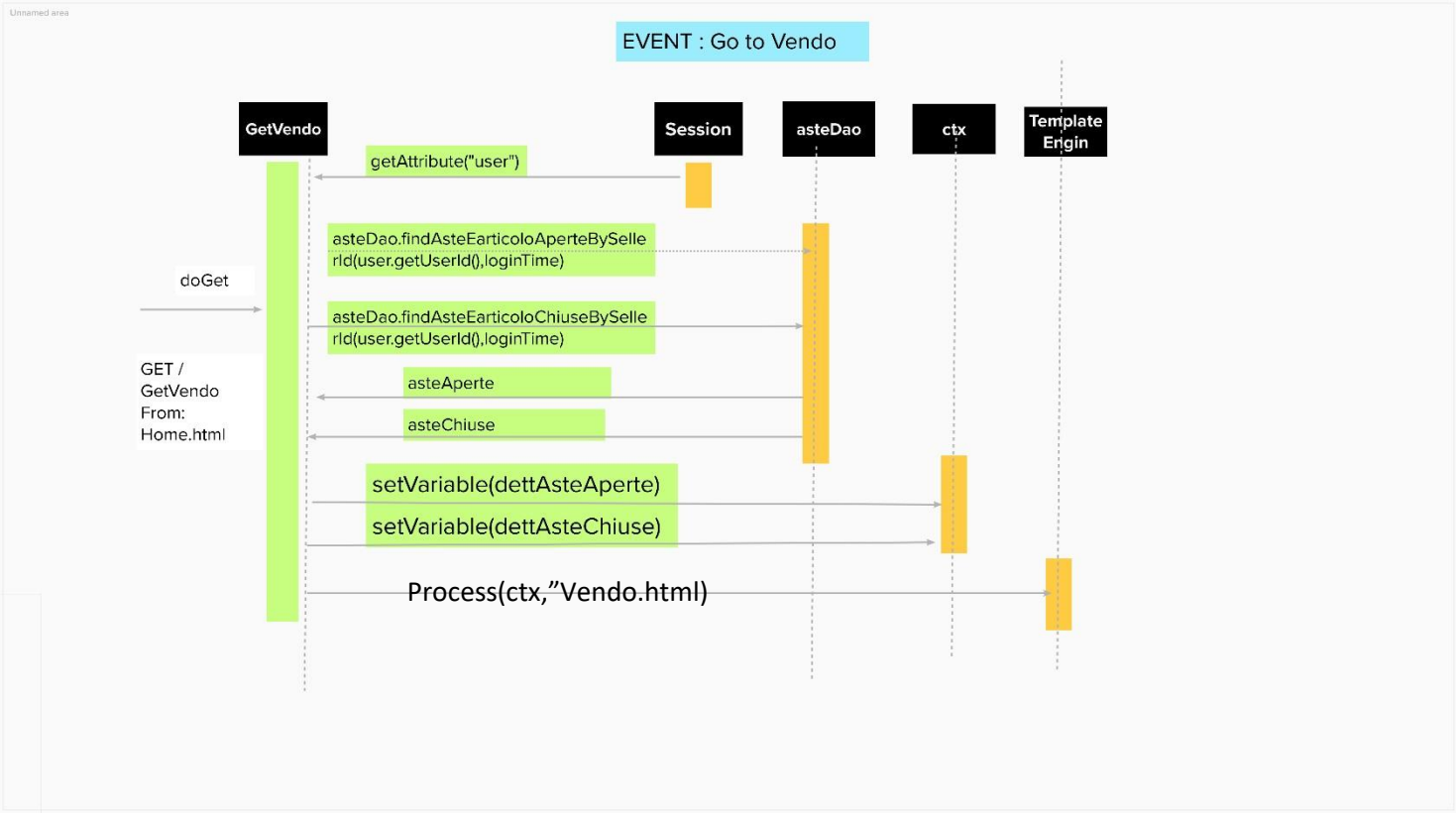
Components

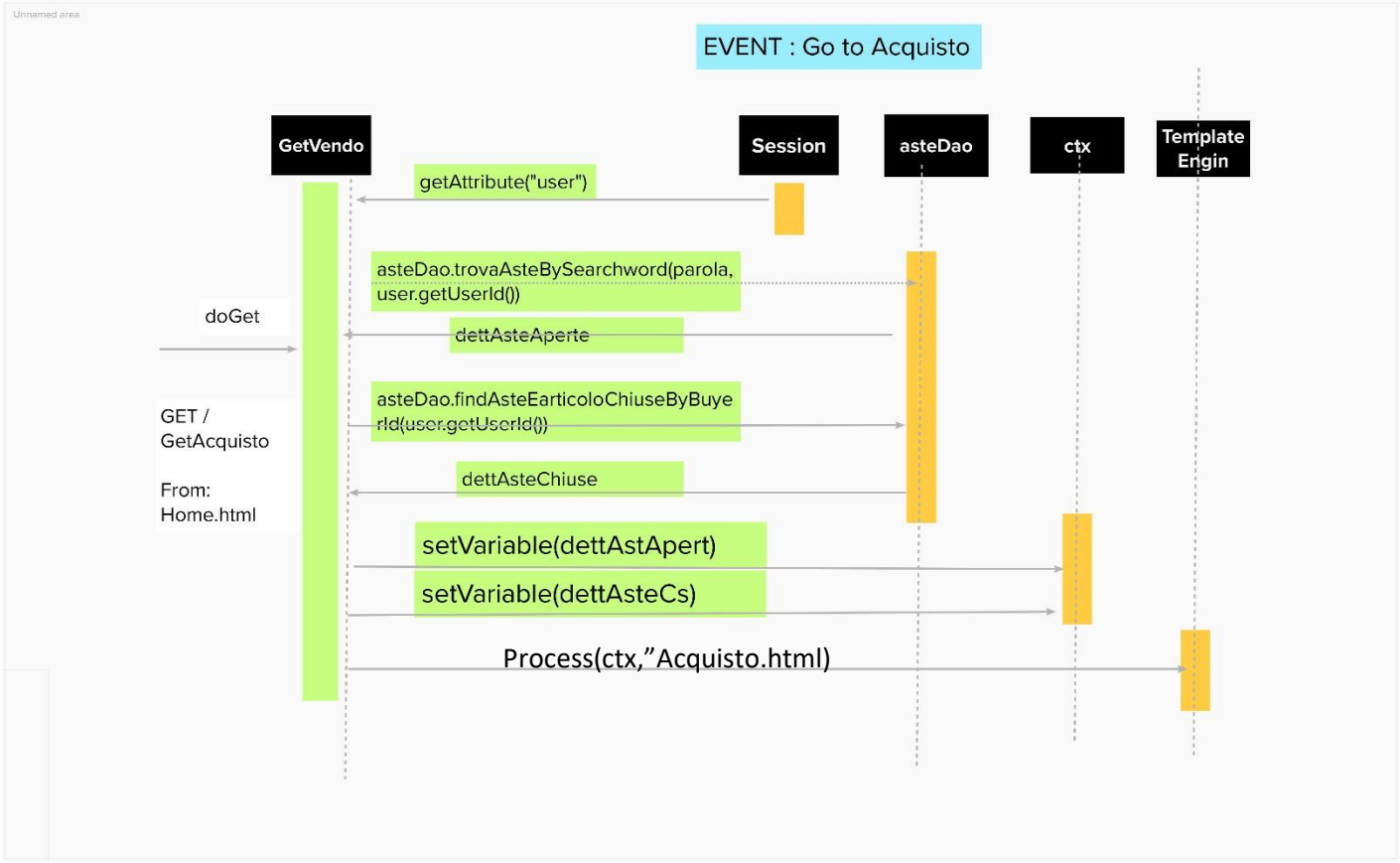
- **Model objects(Beans)**
  - User
  - Asta
  - Offerta
- **Data Access Objects(Classes)**
  - **UserDAO** ○ checkCredentials(username,password) ○ findUserById(userId)
  - **AsteDAO** ○ **findAsteChiuseBySellerId(userId)** ○ **findAsteAperteBySellerId(userId)** ○ **findAsteEarticoloAperteBySellerId(userId, tempoLogin)**
    - **findAsteEarticoloChiuseBySellerId(userId, tempoLogin)**
    - **findAstaById(idAsta)** ○ **findAstaEarticoloByIdAsta(idAsta)** ○ **findAstaEarticoloCsByIdAsta(idAsta)** ○ **trovaAsteBySearchword(parola, userId)** ○ **findAsteEarticoloChiuseByBuyerId(userId)** ○ **changeAstaStatus(idAsta, vincitoreAsta, prezzoAggiudica, astaStatus)**
    - **creaAsta(codice, nome, descrizione, image,float prezzoIniziale, rialzoMinimo, scadenza, sellerId)**
  - **OffertaDAO**
    - **findOfferteByIdAsta(idasta)** ○ **findTutteOfferteByIdAsta(idAsta)** ○ **findOffertaVincente(idasta)** ○ **creaOfferta(ammontareOfferta, dataOraOfferta, idAsta, idOfferente)**

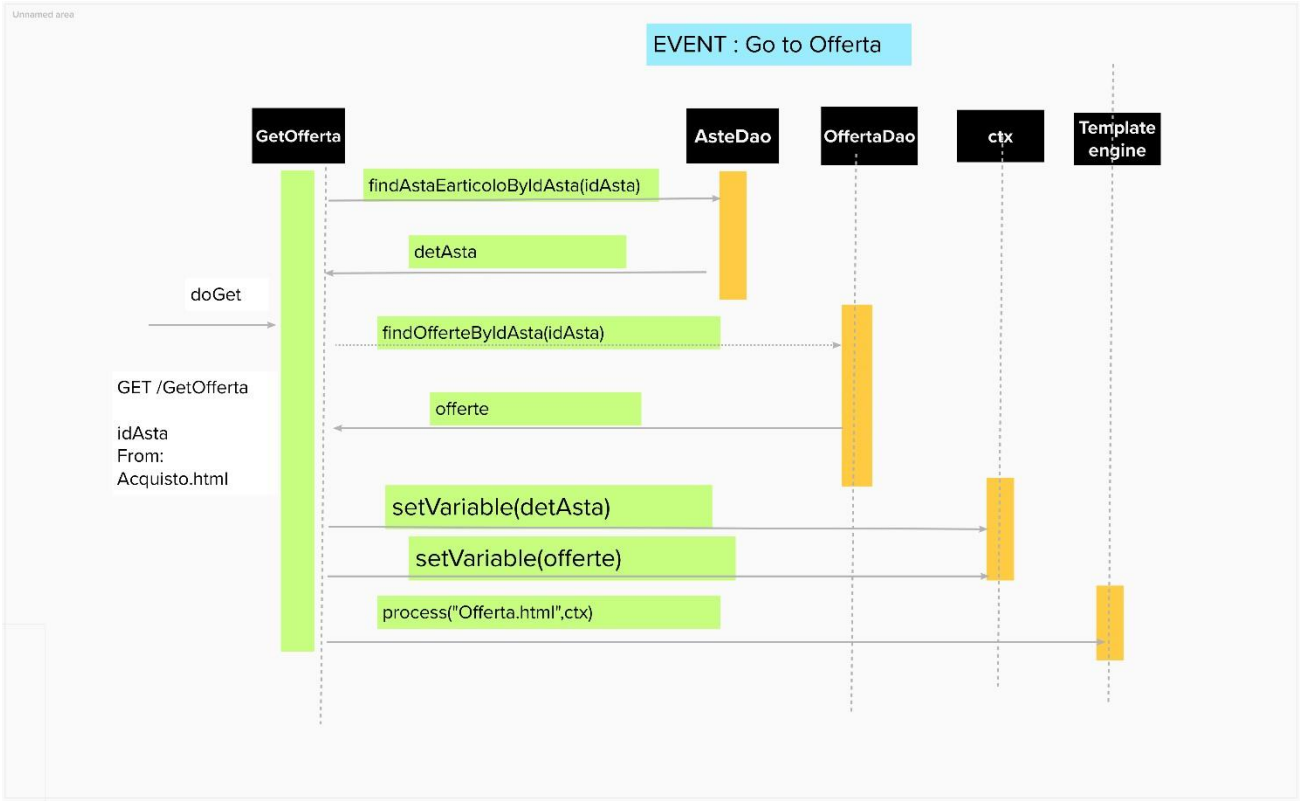
- **Controllers(servlets) - checkLogin**
  - GoToHomepage
  -NuovaAsta
  -ChiudiAsta
  -GetAcquisto
  -GetVendo
  -TrovaAsteConSearchword
  -GetOfferta
  -FaiOfferta
  -Logout

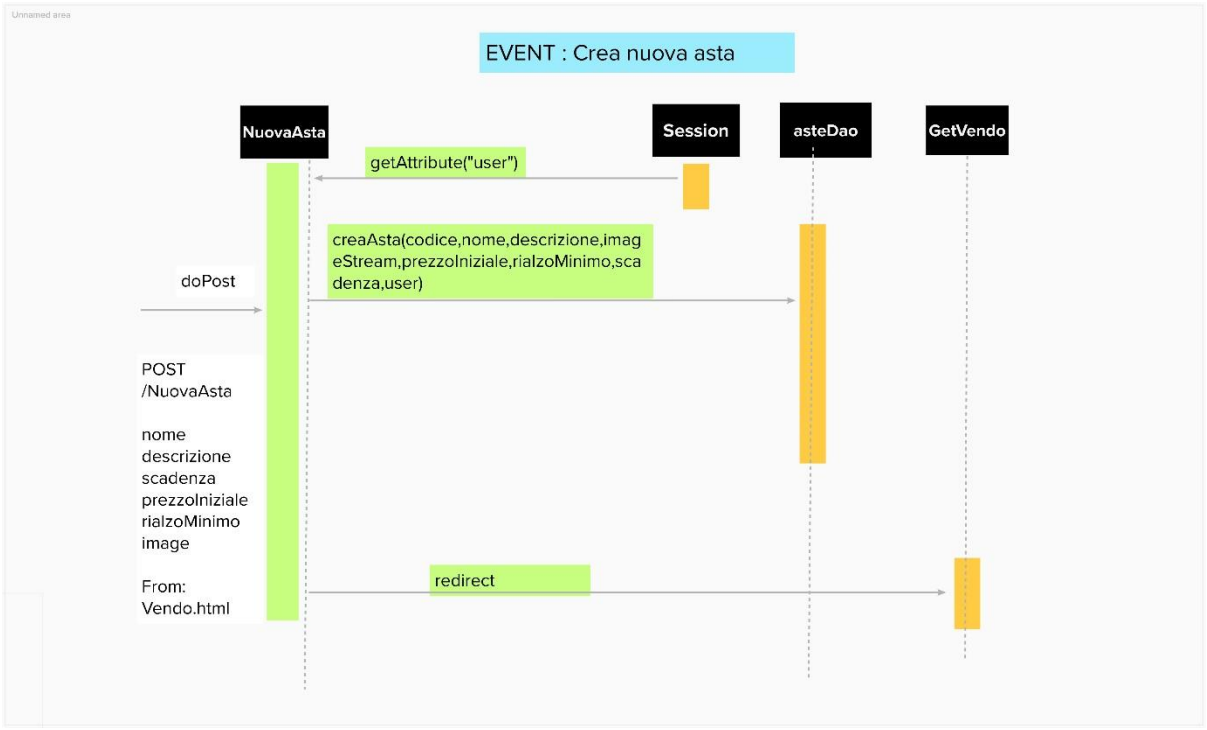
- **Views(Templates)**
  - Login
  -Home
  -Vendo
  -Acquisto
  -Offerta

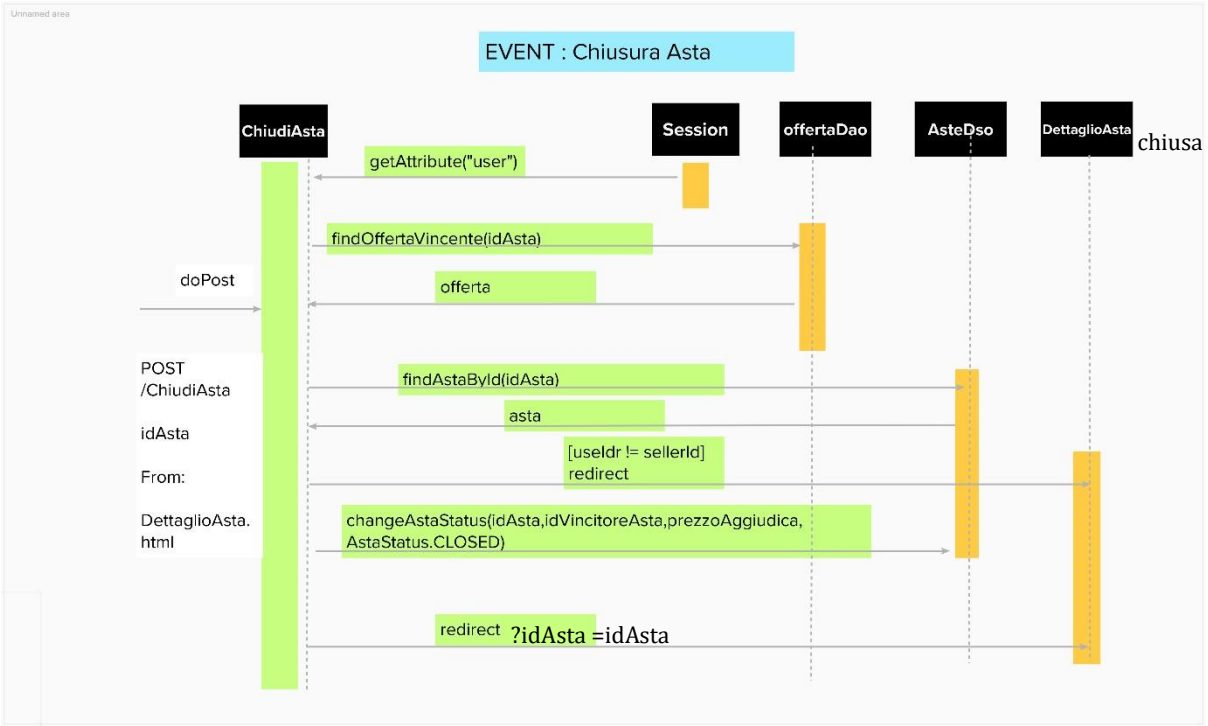

EVENT : Login
[Sequence name]
CheckLogin
UserDAO
index.html
Session
GoToHome
new UserDao
checkCredentials()
doPost
user == null
SCENARIO 1
user == null
redirect
POST /CheckLogin
usrname password
From :index.html
user != null
setAttribute("user",user)
user!= null redirect

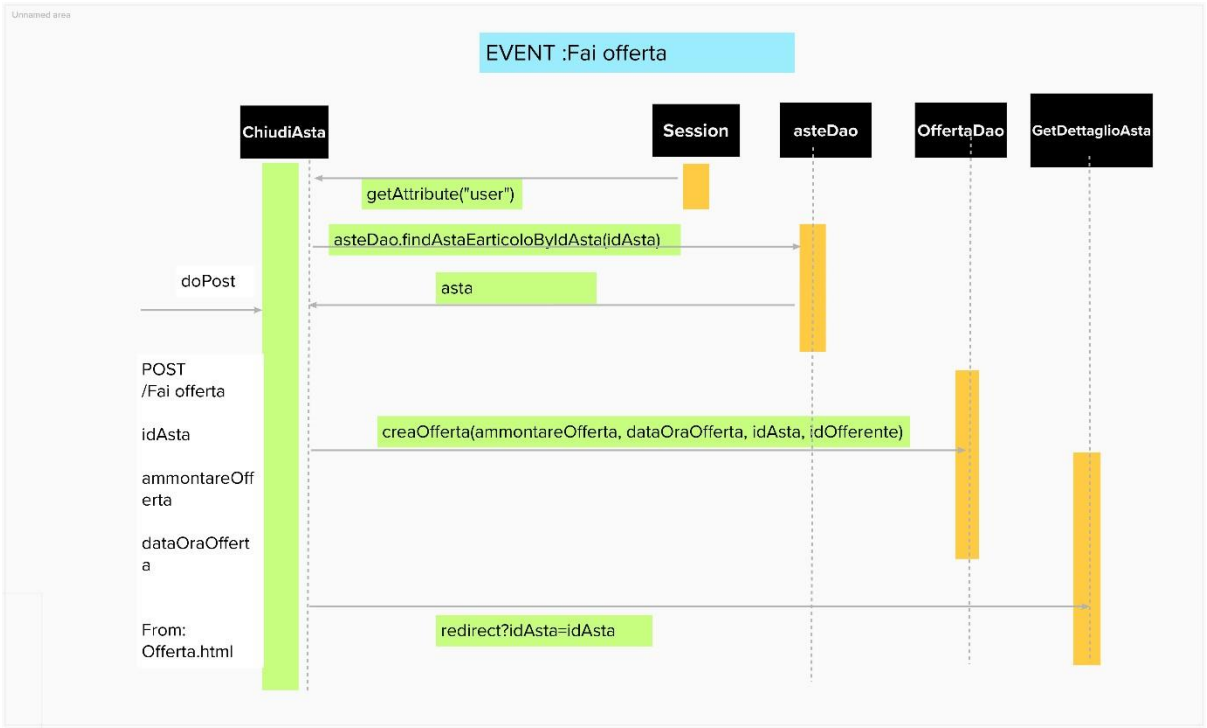

EVENT : Go to Vendo
GetVendo
Session
asteDao
ctx
Template Engin
getAttribute("user")
asteDao.findAsteEarticoloAperteBySelle
rId(user.getUserId(),loginTime)
doGet
asteDao.findAsteEarticoloChiuseBySelle
rId(user.getUserId(),loginTime)
GET /
GetVendo
From:
Home.html
asteAperte
asteChiuse
setVariable(dettAsteAperte)
setVariable(dettAsteChiuse)
Process(ctx,"Vendo.html)

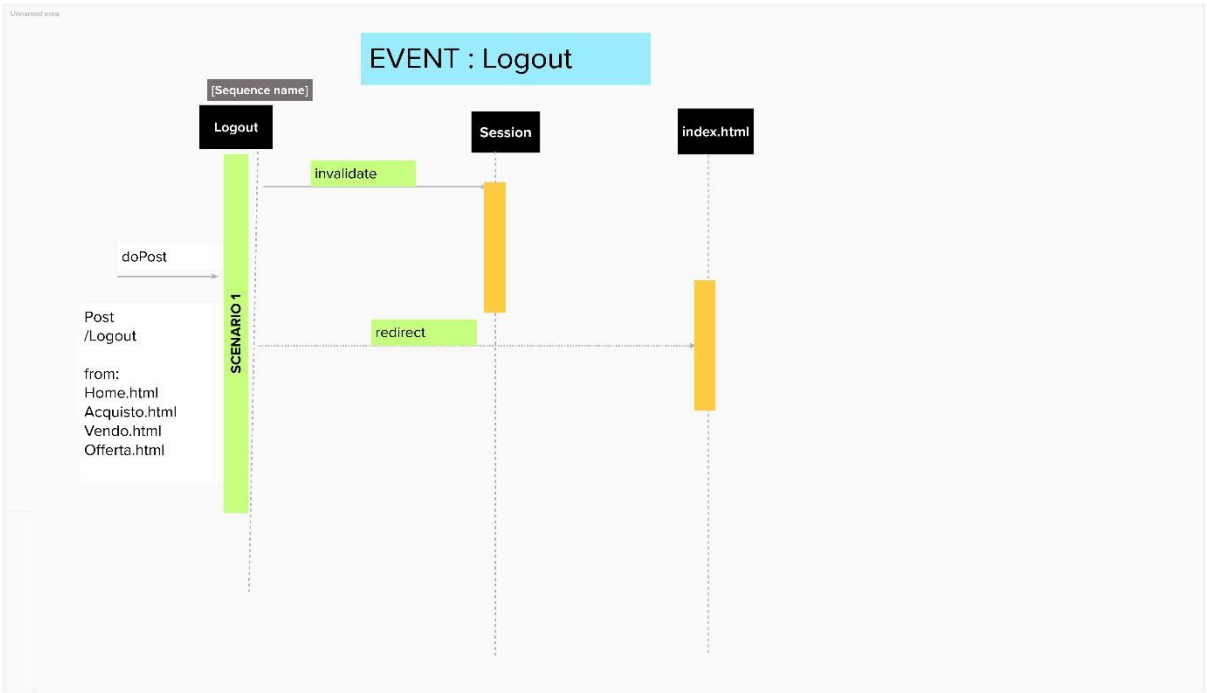

Unnamed area
EVENT : Go to Acquisto
GetVendo
Session
asteDao
ctx
Template Engin
getAttribute("user")
asteDao.trovaAsteBySearchword(parola, user.getUserId())
doGet
dettAsteAperte
asteDao.findAsteEarticoloChiuseByBuyerId(user.getUserId())
GET / GetAcquisto
From: Home.html
dettAsteChiuse
setVariable(dettAstApert)
setVariable(dettAsteCs)
Process(ctx,"Acquisto.html)


Unnamed area
EVENT : Go to Offerta
GetOfferta
AsteDao
OffertaDao
ctx
Template engine
findAstaEarticoloByIdAsta(idAsta)
detAsta
doGet
findOfferteByIdAsta(idAsta)
GET /GetOfferta
idAsta
From: Acquisto.html
offerte
setVariable(detAsta)
setVariable(offerte)
process("Offerta.html",ctx)


Unnamed area
EVENT : Crea nuova asta
NuovaAsta
Session
asteDao
GetVendo
getAttribute("user")
creaAsta(codice,nome,descrizione,imag eStream,prezzoIniziale,rialzoMinimo,sca denza,user)
doPost
POST /NuovaAsta
nome
descrizione
scadenza
prezzoIniziale
rialzoMinimo
image
From: Vendo.html
redirect


Unnamed area
EVENT : Chiusura Asta
ChiudiAsta
Session
offertaDao
AsteDso
DettaglioAsta
chiusa
getAttribute("user")
findOffertaVincente(idAsta)
doPost
offerta
POST /ChiudiAsta
idAsta
From:
DettaglioAsta. html
findAstaById(idAsta)
asta
[useIdr != sellerId] redirect
changeAstaStatus(idAsta,idVincitoreAsta,prezzoAggiudica, AstaStatus.CLOSED)
redirect ?idAsta =idAsta


EVENT :Fai offerta
ChiudiAsta
Session
asteDao
OffertaDao
GetDettaglioAsta
getAttribute("user")
asteDao.findAstaEarticoloByIdAsta(idAsta)
doPost
asta
POST
/Fai offerta
idAsta
ammontareOff
erta
dataOraOffert
a
From:
Offerta.html
creaOfferta(ammontareOfferta, dataOraOfferta, idAsta, idOfferente)
redirect?idAsta=idAsta

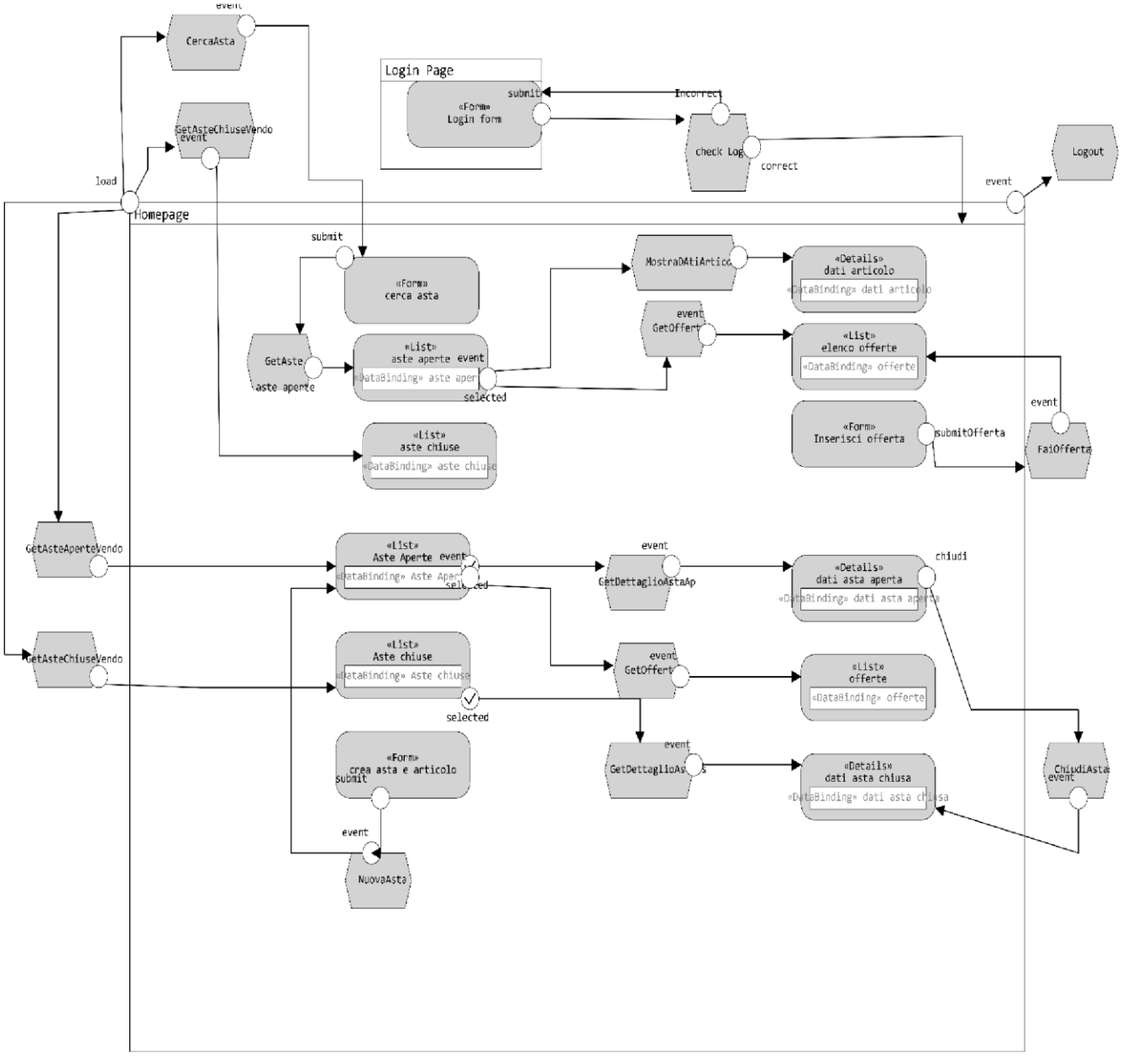

EVENT : Logout
[Sequence name]
Logout
Session
index.html
invalidate
doPost
SCENARIO 1
Post
/Logout
from:
Home.html
Acquisto.html
Vendo.html
Offerta.html
redirect

Versione con JavaScript

Si realizzi un'applicazione client server web che modifica le specifiche precedenti come segue:

- 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.

Application design



- **Server side: DAO & model objects**
- **Model objects(Beans)**

- User
- Asta
- Offerta

- **Data Access Objects(Classes)**

- UserDAO

- checkCredentials(username,password)
- findUserById(userId)

- AsteDAO

- **findAsteChiuseBySellerId(userId)**
- **findAsteAperteBySellerId(userId)**
- **findAsteEarticoloAperteBySellerId(userId, tempoLogin)**
- **findAsteEarticoloChiuseBySellerId(userId, tempoLogin)**
- **findAstaById(idAsta)**
- **findAstaEarticoloByIdAsta(idAsta)**
- **findAstaEarticoloCsByIdAsta(idAsta)**
- **trovaAsteBySearchword(parola, userId)**
- **findAsteEarticoloChiuseByBuyerId(userId)**
- **changeAstaStatus(idAsta, vincitoreAsta, prezzoAggiudica, astaStatus)**
- **creaAsta(codice, nome, descrizione, image,float prezzoIniziale, rialzoMinimo, scadenza, sellerId)**

- OffertaDAO

- **findTutteOfferteByIdAsta(idAsta)**
- **findOffertaVincente(idasta)**
- **creaOfferta(ammontareOfferta, dataOraOfferta, idAsta, idOfferente)**

- **Controllers(servlets)**

  - checkLogin
  - cercaAsta
  -NuovaAsta
  -ChiudiAsta
  -GetAcquisto
  -GetAsteAperteVendo
  -GetAsteChiuseVendo
  GetDettaglioAstaAp
  GetDettaglioAstaCs
  -GetOfferta
  -FaiOfferta
  MostraDatiArticolo
  -Logout

    - **Viste e componenti**
    - **Pagina Home**
    - **Lista aste aperte acquisto**
    - **Lista aste aggiudicate acquisto**
    - **Form ricerca aste**

      Modulo inserimento parola di ricerca
    - **Dato asta aperta**
    - **Lista offerte**
    - **Form inserimento offerta**

      Modulo inserimento ammontare offerta
    - **Lista aste aperte Vendo**
    - **Lista aste chiuse Vendo**
    - **Form creazione asta e articolo**
    - **Modulo inserimento nome**

- **Modulo inserimento descrizione**
- **Modulo inserimento prezzoIniziale**
- **Modulo inserimento rialzoMinimo**
- **Modulo inserimento scadenza**
- **Modulo inserimento immagine**
- **Dettaglio Asta aperta**
- **Lista Offerte**
- **Bottoni chiusura asta**
- **Dettaglio Asta chiusa**
- **Dettaglio spese**

MessaggioBenvenuto

- **Eventi e azioni – Login**
- **Verifica credenziali**

– **Click su submit di nuova asta**
  - **Creazione asta**

– **Click su submit di faiOfferta**
  - **Sottomissione offerta**

– **Click su bottone Vendo**
  - **Mostra pagina Vendo**

– **Click su bottone Acquisto**
  - **Mostra pagina Acquisto**

– **Selezione asta da Lista**
  - **Mostra dettagli asta**

– **Invio form offerta**
  - **Aggiunta offerta all'asta**

– **Invio form NuovaAsta**
  - **Aggiunta nuova asta e articolo**

– **Click bottone chiusura**
  - **Cambio stato asta a chiusa – Logout**
  - **logout**

Client side: view & view component

- **Index**
  - **Login form**
    - **Gestione del submit e degli errori**

Home

Lista aste aperte acquisto

- **update(): riceve dati server e aggiorna la lista** **Lista aste aggiudicate acquisto**
- **show(): richiede al server i dati dell'elenco aste**
- **update(): riceve dati server e aggiorna la lista**

Form ricerca aste

- **registerClick(): associa al componente la funzione per gestire ricerca**

Dato articolo asta aperta

- **show(): richiede al server i dati dell'asta**
- **update(): riceve dati server e aggiorna la lista registerEvents(): associa al componente la funzione per fare offerte**

Form inserimento offerta

Lista offerte Acquisto

- **show(): richiede al server i dati dell'elenco offerte**
- **update(): riceve dati server e aggiorna la lista** **Lista aste aperte Vendo**
- **show(): richiede al server i dati dell'elenco aste**
- **update(): riceve dati server e aggiorna la lista** **Lista aste chiuse Vendo**
- **show(): richiede al server i dati dell'elenco aste**

- **update(): riceve dati server e aggiorna la lista**

Form creazione asta e articolo

- **associa al componente la funzione per gestire creazione dell'asta**

Dettaglio Asta aperta

- **show(): richiede al server i dati dell'asta**
- **registerEvents(): associa al componente la funzione per chiudere l'asta**
- **update(): riceve dati server e aggiorna l'asta**

Bottoni chiusura asta

Lista Offerte

- **show(): richiede al server i dati dell'elenco offerte**
- **update(): riceve dati server e aggiorna la lista**

Dettaglio Asta chiusa

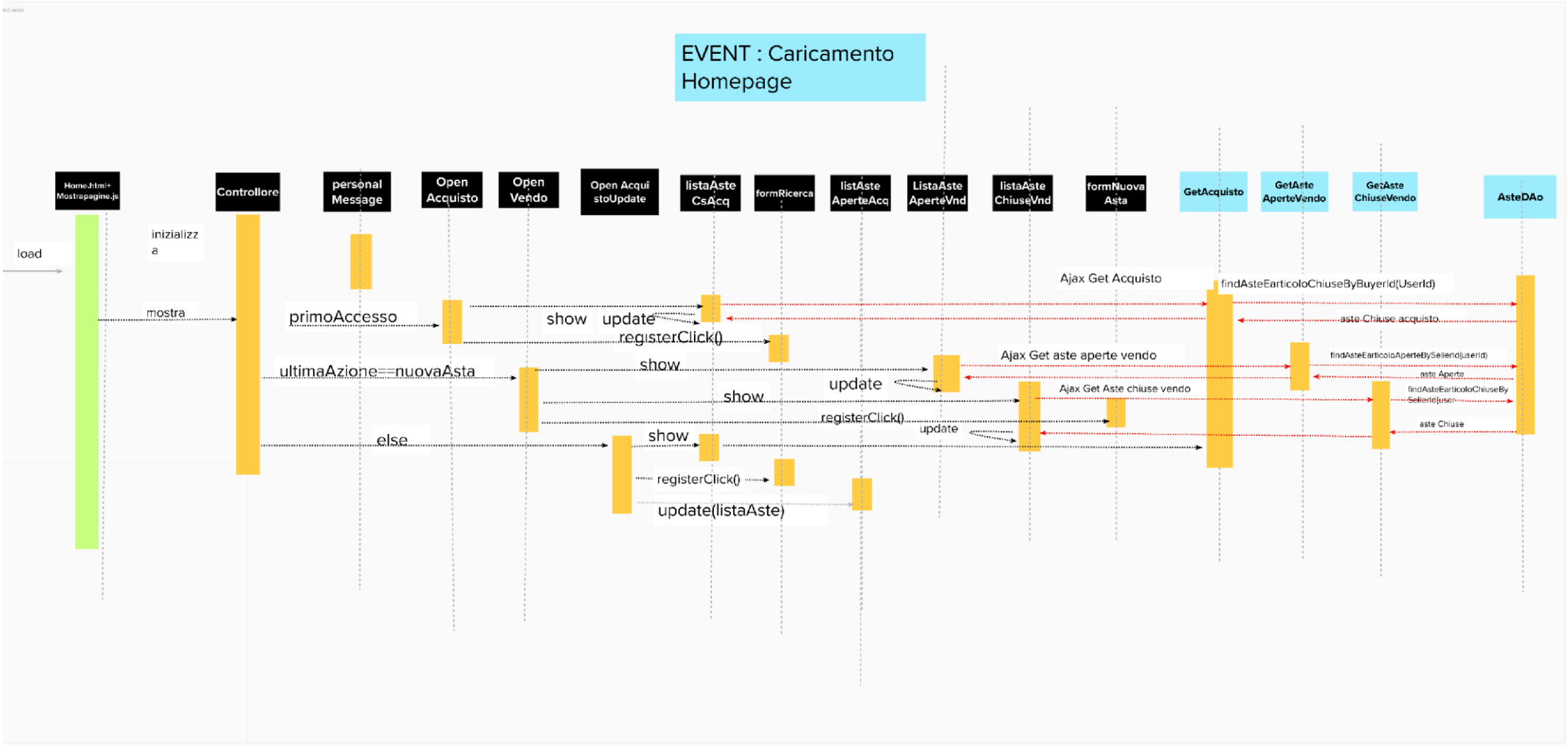
- **show(): richiede al server i dati dell'asta**
- **update(): riceve dati server e aggiorna l'asta**
-

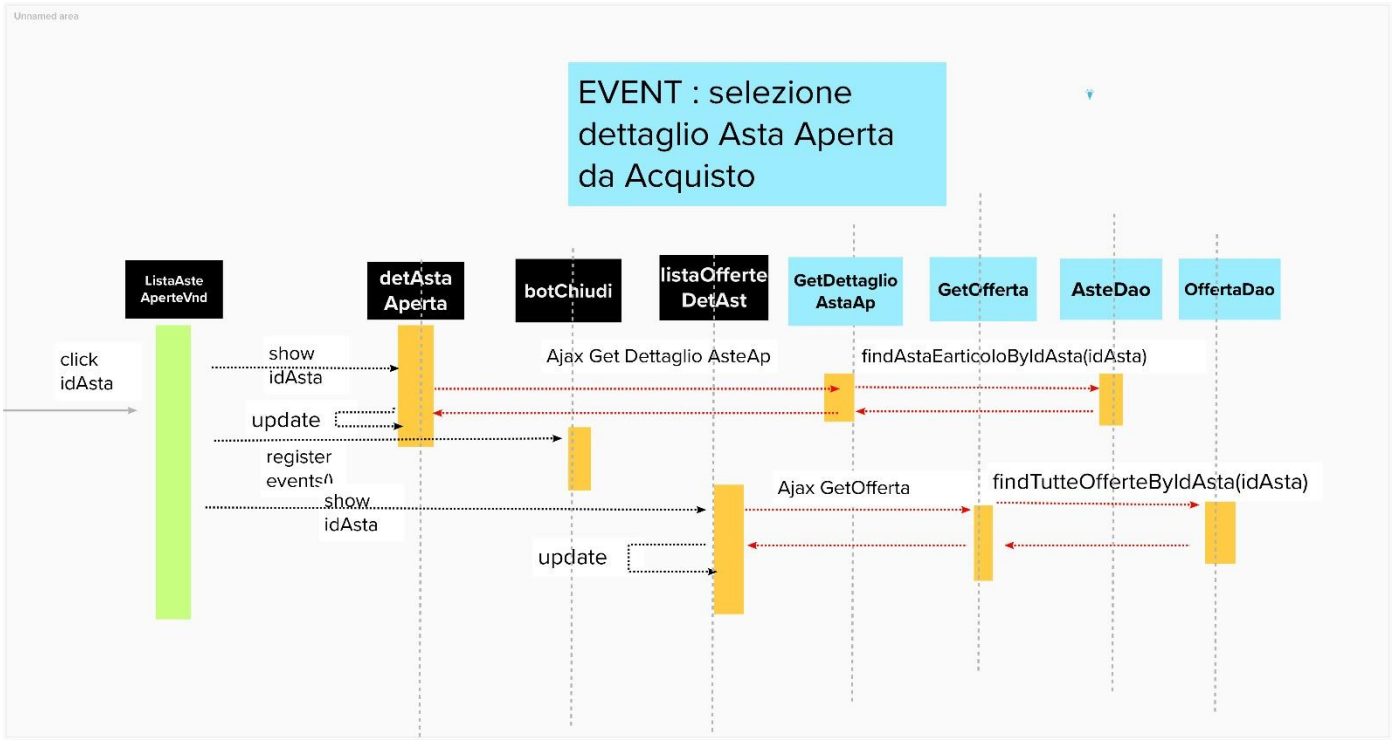
Gestione del ciclo di vita

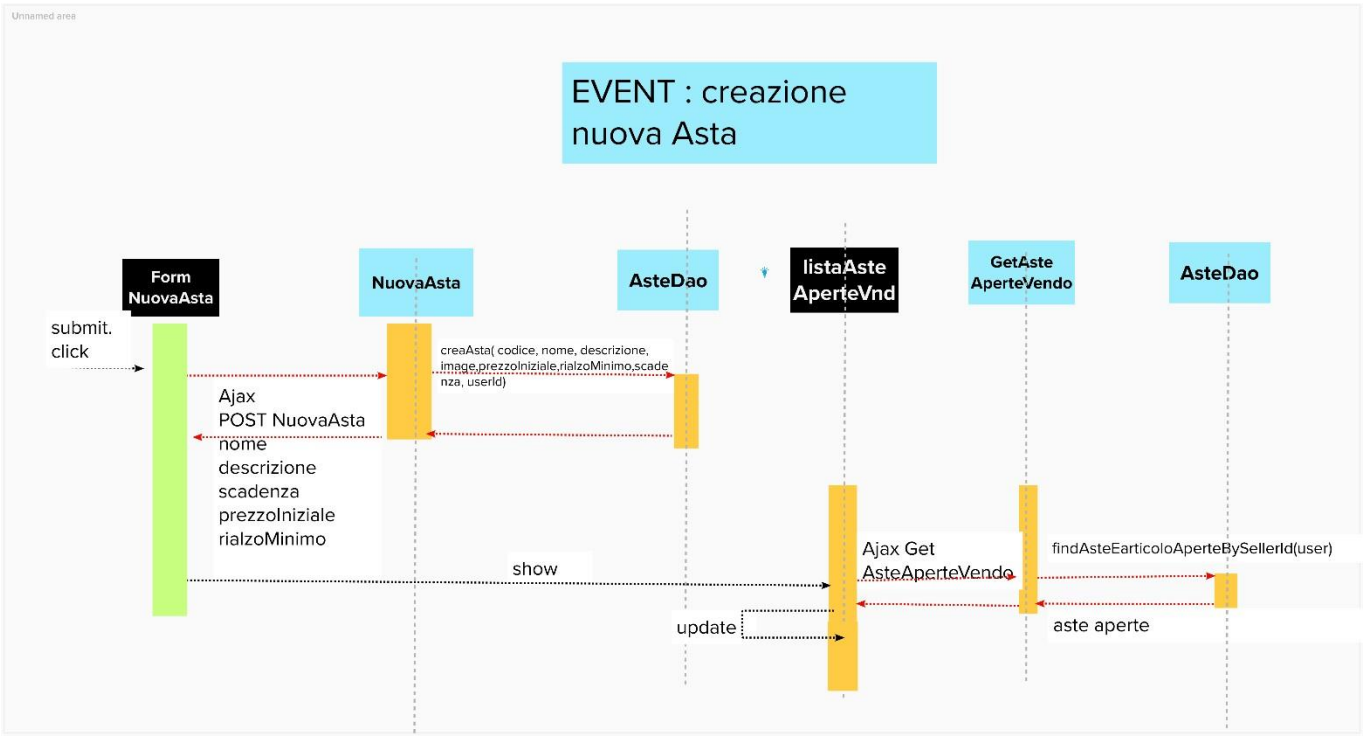
Il componente Controllore gestisce il ciclo di vita con le funzioni

- **Inizializza : per creare e inizializzare i componenti dell'interfaccia.**
- **Mostra : reperisce e mostra i contenuti prima accedendo alle funzioni OpenVendo ,OpenAcquisto o OpenAcquisto(listaAsteViste) a seconda dei cookie salvati.**

Queste tre funzioni mostrano o nascondono le componenti a seconda della vista che si vuole far vedere.


EVENT : Caricamento Homepage
Home.html+ Mostrapagine.js
Controllore
personal Message
Open Acquisto
Open Vendo
Open AcquistoUpdate
listaAste CsAcq
formRicerca
listAste AperteAcq
ListaAste AperteVnd
listaAste ChiuseVnd
formNuova Asta
GetAcquisto
GetAste AperteVendo
GetAste ChiuseVendo
AsteDAo
load
inizializza
mostra
primoAccesso
show
update
registerClick()
ultimaAzione==nuovaAsta
show
update
show
registerClick()
update
else
show
registerClick()
update(listaAste)
Ajax Get Acquisto
findAsteEarticoloChiuseByBuyerId(UserId)
aste Chiuse acquisto
Ajax Get aste aperte vendo
findAsteEarticoloAperteBySellerId(userId)
aste Aperte
Ajax Get Aste chiuse vendo
findAsteEarticoloChiuseBy SellerId(user
aste Chiuse


Unnamed area
EVENT : selezione dettaglio Asta Aperta da Acquisto
ListaAste AperteVnd
detAsta Aperta
botChiudi
listaOfferte DetAst
GetDettaglio AstaAp
GetOfferta
AsteDao
OffertaDao
click idAsta
show idAsta
Ajax Get Dettaglio AsteAp
findAstaEarticoloByIdAsta(idAsta)
update
register events
show idAsta
Ajax GetOfferta
findTutteOfferteByIdAsta(idAsta)
update


Unnamed area
EVENT : creazione nuova Asta
Form NuovaAsta
NuovaAsta
AsteDao
listaAste AperteVnd
GetAste AperteVendo
AsteDao
submit. click
creaAsta(codice, nome, descrizione, image,prezzoIniziale,rialzoMinimo,scadenza, userId)
Ajax POST NuovaAsta
nome
descrizione
scadenza
prezzoIniziale
rialzoMinimo
show
Ajax Get AsteAperteVendo
findAsteEarticoloAperteBySellerId(user)
update
aste aperte