Nella progettazione delle seguenti classi sono stati applicati dei design pattern:

* Asta\_Factory
  + Design pattern: Factory;
  + Classi coinvolte: Asta\_Controller, Asta\_Factory, Asta, Oggetto, Servizio\_Email;
  + Descrizione: la factory si occuperà di istanziare l’asta assegnando il nome, data\_inzio, durata ed aggiungendo i vari oggetti, utilizzando le informazioni che il controller riceve dall’interfaccia. Inoltre, manda una email a tutti gli utenti registrati per notificare la creazione di un’asta.
* Offerte\_Listener
  + Design pattern: Observer;
  + Classi coinvolte: Pagina\_Asta, Offerte\_Listener, Asta;
  + Descrizione: Pagina\_Asta, che implementa l’interfaccia Offerte\_Listener, sarà “in ascolto” sulla proprietà offerte delle aste; ogni volta che un utente farà una nuova offerta, la pagina relativa a quell’asta si aggiornerà per mostrare la nuova offerta.
* Pagamento\_Adapter
  + Design pattern: Adapter;
  + Classi coinvolte: Asta, Pagamento\_Adapter, Metodo\_Di\_Pagamento;
  + Descrizione: l’adapter invierà richieste all’endpoint specificato per verificare le informazioni del metodo di pagamento di un nuovo utente e per effettuare l’addebito quando un utente vince un’asta. Le specifiche della richiesta dipenderanno dal tipo di metodo di pagamento, e saranno definite nelle classi che implementeranno questa interfaccia.
* Vettore\_Adapter
  + Design pattern: Adapter;
  + Classi coinvolte: Asta, Vettore \_Adapter, Oggetto;
  + Descrizione: l’adapter, alla chiusura dell’asta e dopo che è stato effettuato l’addebito al vincitore, effettuerà una richiesta all’endpoint specificato per preparare la spedizione di un determinato oggetto; inoltre, dopo che è stata creata una spedizione, farà delle richieste per ottenere informazioni sulla spedizione. Le specifiche della richiesta dipenderanno dal vettore di spedizione scelto, e saranno definite nelle classi che implementeranno questa interfaccia.