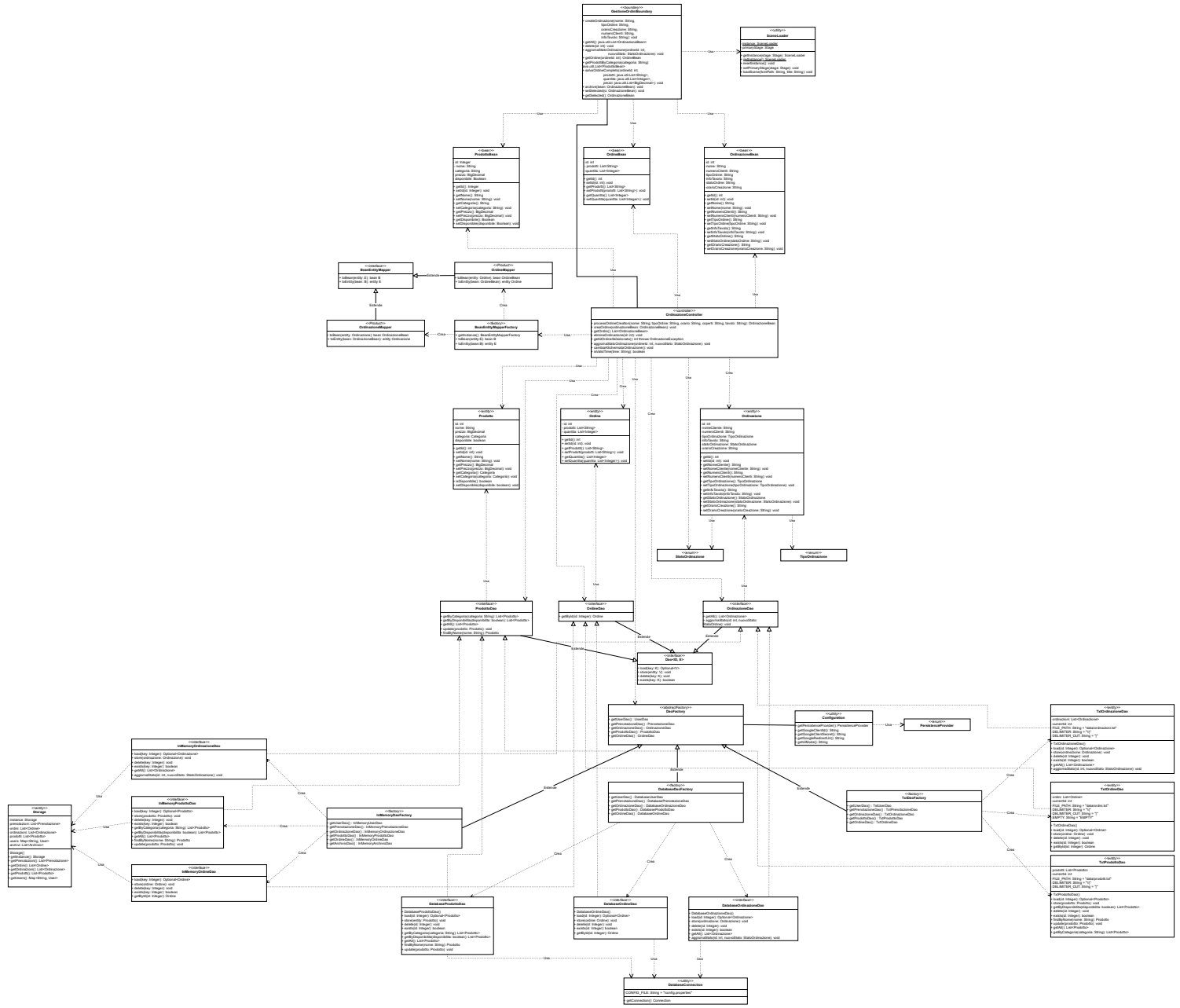
<<boundary>>
OrdineOrdineBoundary
+ createOrdinazione(nome: String, tipoOnline: String, statoOrdine: String, numeroClient: String, infoTavolo: String) : void
+ getAll() : java.util.List<OrdinazioneBean>
+ delete(id: int) : void
+ aggiornaStato(numeroordine(ordine): int, nuovoStato: StatoOrdinazione) : void
+ getOrdine(ordineId: int) : OrdineBean
+ getOrdineByCategoria(categoria: String) : bean.dt.ListProdottiBean
+ saveOrdineCompleto(ordineId: int, prodotti: java.util.List<String>, quantita: java.util.List<Integer>, numeroTavolo: String, commento: String) : void
+ archiviaOrdine(ordineId: int) : void
+ setStateOrdine(ordineId: int) : void
+ getSelected() : OrdinazioneBean
<<utility>>
ScreenLoader
+ screen_Scanner: static
primaryStage: Stage
+ getPrimaryStage(): Stage, Scanner, static
+ setScanner(): Scanner, static
+ initialize() : void
+ setPrimaryStage(stage: Stage) : void
+ loadScreen(fxmlPath: String, title: String) : void
<<bean>>
ProdottoBean
id: Integer
name: String
categoria: String
prezzo: BigDecimal
disponibile: Boolean
+ getId() : Integer
+ setId(id: Integer) : void
+ getName() : String
+ setName(name: String) : void
+ getCategoria() : String
+ setCategoria(categoria: String) : void
+ getPrezzo() : BigDecimal
+ setPrezzo(prezzo: BigDecimal) : void
+ getDisponibile() : Boolean
+ setDisponibile(disponibile: Boolean) : void
<<bean>>
OrdineBean
id: int
prodotti: List<String>
quantita: List<Integer>
+ getId() : int
+ setId(id: int) : void
+ getProdotti() : List<String>
+ setProdotti(prodotti: List<String>) : void
+ getQuantita() : List<Integer>
+ setQuantita(quantita: List<Integer>) : void
<<bean>>
OrdinazioneBean
id: int
nome: String
numeroClient: String
tipoOnline: String
infoTavolo: String
statoOrdine: String
orariCreazione: String
+ getId() : int
+ setId(id: int) : void
+ getNome() : String
+ setNome(nome: String) : void
+ getNumeroClient() : String
+ setNumeroClient(numeroClient: String) : void
+ getTipoOnline() : String
+ setTipoOnline(tipoOnline: String) : void
+ getInfoTavolo() : String
+ setInfoTavolo(infoTavolo: String) : void
+ getStatoOrdine() : String
+ setStatoOrdine(statoOrdine: String) : void
+ getOrariCreazione() : String
+ setOrariCreazione(orariCreazione: String) : void
<<interface>>
BeanEntityMapper
+ toBean(entity: E) : bean B
+ toEntity(bean: B) : entity E
Extends
<<Product>>
OrdineMapper
+ toBean(entity: Ordine) : bean OrdineBean
+ toEntity(bean: OrdineBean) : entity Ordine
Extends
<<Product>>
OrdinazioneMapper
+ toBean(entity: Ordinazione) : bean OrdinazioneBean
+ toEntity(bean: OrdinazioneBean) : entity Ordinazione
Crea
<<factory>>
BeanEntityMapperFactory
+ getInstance() : BeanEntityMapperFactory
+ toBean(entity E) : bean B
+ toEntity(bean B) : entity E
Crea
Usa
<<controller>>
OrdinazioneController
+ processOrdineCreation(nome: String, tipoOnline: String, statoOrdine: String, numeroClient: String, infoTavolo: String) : void
+ createOrdinazione(ordinazioneBean: OrdinazioneBean) : void
+ getOrdine() : List<OrdinazioneBean>
+ deleteOrdine(ordineId: int) : void
+ getAllOrdineRelationship() : int throws OrdinazioneException
+ aggiornaStatoOrdinazione(ordineId: int, nuovoStato: StatoOrdinazione) : void
+ setOrdineCompletato(ordineId: int) : void
+ isValidTime(time: String) : boolean
<<entity>>
Prodotto
id: int
nome: String
prezzo: BigDecimal
categoria: Categoria
disponibile: boolean
+ getId() : int
+ setId(id: int) : void
+ getNome() : String
+ setNome(nome: String) : void
+ getPrezzo() : BigDecimal
+ setPrezzo(prezzo: BigDecimal) : void
+ getCategoria() : Categoria
+ setCategoria(categoria: Categoria) : void
+ isDisponibile() : boolean
+ setDisponibile(disponibile: boolean) : void
<<entity>>
Ordine
id: int
prodotti: List<String>
quantita: List<Integer>
+ getId() : int
+ setId(id: int) : void
+ getProdotti() : List<String>
+ setProdotti(prodotti: List<String>) : void
+ getQuantita() : List<Integer>
+ setQuantita(quantita: List<Integer>) : void
<<entity>>
Ordinazione
id: int
nome: String
numeroClient: String
tipoOnline: TipoOrdinazione
infoTavolo: String
statoOrdinazione: StatoOrdinazione
orariCreazione: String
+ getId() : int
+ setId(id: int) : void
+ getNome(nome): String
+ setNome(nome: String): void
+ getNumeroClient(): String
+ setNumeroClient(numeroClient: String): void
+ getTipoOrdinazione(): TipoOrdinazione
+ setTipoOrdinazione(tipoOrdinazione: TipoOrdinazione): void
+ getInfoTavolo(): String
+ setInfoTavolo(infoTavolo: String): void
+ getStatoOrdinazione() : StatoOrdinazione
+ setStatoOrdinazione(statoOrdinazione: StatoOrdinazione): void
+ getOrariCreazione() : String
+ setOrariCreazione(orariCreazione: String): void
<<enum>>
StatoOrdinazione
<<enum>>
TipoOrdinazione
<<interface>>
ProdottoDao
+ getByCategoria(categoria: String) : List<Prodotto>
+ getByDisponibile(disponibile: boolean) : List<Prodotto>
+ getAll() : List<Prodotto>
+ update(prodotto: Prodotto) : void
+ findByName(name: String) : Prodotto
<<interface>>
OrdineDao
+ getById(id: Integer) : Ordine
<<interface>>
OrdinazioneDao
+ getAll() : List<Ordinazione>
+ aggiornaStato(id: int, nuovoStato: StatoOrdine) : void
Extends
Extends
Extends
<<interface>>
StartDB_Dr
+ load(key: K) : Optional<V>
+ store(entity: V) : void
+ delete(key: K) : void
+ exists(key: K) : boolean
<<abstractFactory>>
DaoFactory
+ getUserDao() : UserDao
+ getPrenotazioneDao() : PrenotazioneDao
+ getOrdinazioneDao() : OrdinazioneDao
+ getProdottoDao() : ProdottoDao
+ getOrdineDao() : OrdineDao
<<utility>>
Configuration
getPersistenceProvider(): PersistenceProvider
getGoogleClientId(): String
getGoogleClientSecret(): String
getGoogleRedirectUri(): String
getURLMap(): String
Usa
<<enum>>
PersistenceProvider
<<utility>>
TxOrdinazioneDao
ordinazioni: List<Ordinazione>
currentId: int
FILE_PATH: String = "tabelle/ordinazioni.txt"
DELIMITER: String = "|"
DELIMITER_OUT: String = "|"
+ TxOrdinazioneDao()
+ load(id: Integer): Optional<Ordinazione>
+ store(ordinazione: Ordinazione): void
+ delete(id: Integer): void
+ exists(id: Integer): boolean
+ getAll(): List<Ordinazione>
+ aggiornaStato(id: int, nuovoStato: StatoOrdinazione): void
<<interface>>
InMemoryOrdinazioneDao
+ load(key: Integer): Optional<Ordinazione>
+ store(ordinazione: Ordinazione): void
+ delete(key: Integer): void
+ exists(key: Integer): boolean
+ getAll(): List<Ordinazione>
+ aggiornaStato(id: int, nuovoStato: StatoOrdinazione): void
<<interface>>
InMemoryProdottoDao
+ load(key: Integer): Optional<Prodotto>
+ store(prodotto: Prodotto): void
+ delete(key: Integer): void
+ exists(key: Integer): boolean
+ getByCategoria(categoria: String): List<Prodotto>
+ getByDisponibile(disponibile: boolean): List<Prodotto>
+ getAll(): List<Prodotto>
+ findByName(name: String): Prodotto
+ update(prodotto: Prodotto): void
<<factory>>
DaoFactory
+ getUserDao() : DatabaseUserDao
+ getPrenotazioneDao() : DatabasePrenotazioneDao
+ getOrdinazioneDao() : DatabaseOrdinazioneDao
+ getProdottoDao() : DatabaseProdottoDao
+ getOrdineDao() : DatabaseOrdineDao
<<factory>>
TxDaoFactory
+ getUserDao(): TxUserDao
+ getPrenotazioneDao() : TxPrenotazioneDao
+ getOrdinazioneDao() : TxOrdinazioneDao
+ getProdottoDao() : TxProdottoDao
+ getOrdineDao() : TxOrdineDao
<<entity>>
TxOrdineDao
ordini: List<Ordine>
currentId: int
FILE_PATH: String = "tabelle/ordini.txt"
DELIMITER: String = "|"
DELIMITER_OUT: String = "|"
EMPTY: String = "EMPTY"
+ TxOrdineDao()
+ load(id: Integer): Optional<Ordine>
+ store(ordine: Ordine): void
+ delete(id: Integer): void
+ exists(id: Integer): boolean
+ getAll(key): Integer): Ordine
<<entity>>
TxProdottoDao
prodotti: List<Prodotto>
currentId: int
FILE_PATH: String = "tabelle/prodotti.txt"
DELIMITER: String = "|"
DELIMITER_OUT: String = "|"
+ TxProdottoDao()
+ load(id: Integer): Optional<Prodotto>
+ store(prodotto: Prodotto): void
+ findAll(disponibile: boolean): List<Prodotto>
+ delete(id: Integer): void
+ exists(id: Integer): boolean
+ findByName(name: String): Prodotto
+ update(prodotto: Prodotto): void
+ getAll(): List<Prodotto>
+ getByCategoria(categoria: String): List<Prodotto>
<<interface>>
InMemoryDao
+ load(key: Integer): Optional<Object>
+ store(entity: Object): void
+ delete(key: Integer): void
+ exists(key: Integer): boolean
+ getAll(key): Integer): Object
<<interface>>
InMemoryDaoFactory
+ getUserDao() : InMemoryUserDao
+ getPrenotazioneDao() : InMemoryPrenotazioneDao
+ getOrdinazioneDao() : InMemoryOrdinazioneDao
+ getProdottoDao() : InMemoryProdottoDao
+ getOrdineDao() : InMemoryOrdineDao
+ getArchivioDao() : InMemoryArchivioDao
<<utility>>
Storage
instance: Storage
prenotazioni: List<Prenotazione>
ordini: List<Ordine>
ordinazioni: List<Ordinazione>
prodotti: List<Prodotto>
users: Map<String, User>
archivi: List<Archivio>
+ Storage()
+ getInstance(): Storage
+ getPrenotazioni(): List<Prenotazione>
+ getOrdini(): List<Ordine>
+ getOrdinazioni(): List<Ordinazione>
+ getProdotti(): List<Prodotto>
+ getUsers(): Map<String, User>
Usa
Usa
Usa
Crea
Crea
Crea
Crea
Crea
Crea
Crea
Crea
Crea
Crea
Crea
Crea
<<interface>>
DatabaseProdottoDao
+ DatabaseProdottoDao()
+ load(id: Integer): Optional<Prodotto>
+ store(prodotto: Prodotto): void
+ delete(id: Integer): void
+ exists(id: Integer): boolean
+ getByCategoria(categoria: String): List<Prodotto>
+ getByDisponibile(disponibile: boolean): List<Prodotto>
+ getAll(): List<Prodotto>
+ findByName(name: String): Prodotto
+ update(prodotto: Prodotto): void
<<interface>>
DatabaseOrdineDao
+ DatabaseOrdineDao()
+ load(id: Integer): Optional<Ordine>
+ store(ordine: Ordine): void
+ delete(id: Integer): void
+ exists(id: Integer): boolean
+ getById(key): Integer): Ordine
<<interface>>
DatabaseOrdinazioneDao
+ DatabaseOrdinazioneDao()
+ load(id: Integer): Optional<Ordinazione>
+ store(ordinazione: Ordinazione): void
+ delete(id: Integer): void
+ exists(id: Integer): boolean
+ getAll(): List<Ordinazione>
+ aggiornaStato(id: int, nuovoStato: StatoOrdinazione): void
Usa
Usa
Usa
<<utility>>
DatabaseConnection
CONFIG_FILE: String = "config.properties"
+ getConnection(): Connection