Tessera: C4 Model Documentation

System Context Diagram

Overview:
Tessera is a web application for managing and selling tickets to themed (antique) events, supporting different user roles and integrating with an external payment service.

Diagram:

```
[imperator]          [legatus_augusti]          [legatus_legionis]   [tribunus_laticlavius]
         \                    |                          |                        /
          \                   |                          |                       /
           \                  |                          |                      /
            +----------------------------------------------------------------+
            |                       Tessera System                           |
            |---------------------------------------------------------------|
            | [webapp] <--> [api] <--> [database]                           |
            |          \        |                                           |
            |           \       +--> [payment_system (Stripe, etc.)]        |
            +---------------------------------------------------------------+
           /    |        \              |         |        \              \
 [praefectus...] [tribuni_angusticlavii] [primus_pilus] [centurion] [guest]
                [optio] [tesserarius] [decanus] [miles_gregarius]
```

Key Relationships:

- All user roles interact with Tessera (webapp).
- Tessera (API) communicates with a MySQL database and an external payment system (e.g., Stripe).

Container Diagram

Overview:
Tessera is split into three main containers:

- **Webapp:** Angular SPA for event/ticket viewing and purchase.
- **API:** ASP.NET Core backend for event/ticket management.
- **Database:** MySQL for storing events, tickets, and users.
- **Payment System:** External payment provider.

Diagram:

```
+------------------------------+
|         Users                |
+-----------------------------+
            |
            v
+-----------------------------+
|      Tessera Webapp         |  (Angular SPA)
+-----------------------------+
            |
            v        (REST, JSON/HTTP)
+-----------------------------+
|        Tessera API          |  (ASP.NET Core)
+-----------------------------+
  |                |
  v                v
[MySQL Database]   [Payment System]
```

Relations:

- Users interact with Webapp (different UI per role).
- Webapp calls API (REST).
- API accesses MySQL (SQL), and Payment System (HTTPS).

Component Diagram (API)

Overview:
Key backend components (all in ASP.NET Core):

- `EventController`: Handles event endpoints (`GET /api/events`)
- `TicketController`: Handles ticket endpoints (`POST /api/tickets`)
- `PaymentService`: Integrates with payment system
- `EventService`/`TicketService`: Business logic
- `Repository`: Database access

Diagram:

```
[Webapp] --> [EventController] --> [EventService] --> [Repository] --> [MySQL]
    \           |
     \          v
      \----> [TicketController] --> [TicketService] --> [Repository]
                  |
                  v
           [PaymentService] --> [Payment System]
```

Component Diagram (Webapp)

Overview:
Key frontend components (all in Angular/TypeScript):

- `EventListComponent`: List events
- `TicketPurchaseComponent`: Ticket purchase form
- `AdminDashboardComponent`: Admin UI

Diagram:

```
[User] --> [EventListComponent] --\
        [TicketPurchaseComponent]---> (API)
        [AdminDashboardComponent]---/
```

Each component calls the relevant API endpoints.

Roles Table

Role	Description	Main Permissions/Interfaces Used
Imperator	/	Admin dashboard, all event/ticket actions
Legatus Augusti		Admin dashboard, system settings
Legatus Legionis		Admin/event management
Tribunus Laticlavius		Admin/event management
Praefectus Castrorum		Event organization interface
Tribuni Angusticlavii		Event moderation interface
Primus Pilus	/	Ticket/event moderation
Centurion		Ticket management
Optio	/	Ticket purchase/reservation
Tesserarius		Ticket reservation
Decanus		Ticket purchase
Miles Gregarius		Ticket purchase
Guest		Event listing (read-only)

Key Relationships

- **Users → Webapp:** Interact through different interfaces based on role.
- **Webapp → API:** REST API for all business logic.
- **API → DB:** CRUD operations.
- **API → Payment System:** Payment processing.

External Systems

- **Payment System (Stripe or similar):**
  - Used for payment processing.
  - Accessed by API during ticket purchase.

Technology Stack

- **Frontend:** Angular (TypeScript, HTML/CSS)
- **Backend:** ASP.NET Core (C#)
- **Database:** MySQL
- **External Payment:** Stripe (or similar)

Summary

Tessera provides a robust and role-based system for managing themed events and tickets, with a clear separation of concerns (frontend, backend, database, external payments) and fine-grained access control.