Ticket Trade System

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Projectteam

|  |  |  |  |
| --- | --- | --- | --- |
| **Naam en voornaam** | Volgt BE | Volgt  FE | Functie binnen het team |
| Declercq Hannes | X |  | Backend |
| Butseraen Jim | X |  | Backend |

# Deployment parameters

|  |  |
| --- | --- |
| **TFS of GITHUB?** | GitHub |
| **URL** | <https://github.com/hannesdeclercq/TicketRuilSysteem> |
| **HOSTING HEROKU** | <https://tickettrade.herokuapp.com/> |

# Gebruikerseisen

**Wat verwacht de gebruiker van je toepassing?**

## Front End (n.v.t.)

De front van de applicatie bestaat uit verschillende delen.

* Welkom pagina met een summier maar duidelijk overzicht van actueel beschikbare tickets.
* Pagina die de werking van de applicatie beschrijft. Maak dit visueel, zorg dat dit niet een gewone tekstpagina is
* Pagina waar je uw evenement kan opvragen of zoeken op basis van verschillende criteria ( soort, kost, tijd).
* Overzicht van beschikbare tickets
* Overzicht van jouw tickets
* Overzicht van jouw tickets die je geruild hebt.
* Een aantal statistieken voor de administrator.

## Back End

* Alle data wordt opgeslagen en is voorzien van een timestamp.
* Voorzie een onderhandelmogelijkheid waarbij de verruiler in real-time kan communiceren met de persoon die een ticket aanvraagt.
* Een administrator kan na inloggen een aantal statistieken opvragen: welke regio meest actief, welke type tickets het meest geruild, gemiddelde prijs.

**Wat zijn de drie belangrijkste elementen, die een meerwaarde bieden?**

1. Tickets kunnen aanbieden
2. Evenement kunnen opzoeken
3. Tickets kunnen kopen

# Implementatie –methodiek & technologie

## FrontEnd

|  |  |  |
| --- | --- | --- |
| Technologie | Waarom | Commentaar |
| Bootstrap | Simpel | / |
| Jade | Standaard in Express | Template language |

## BackEnd

|  |  |  |
| --- | --- | --- |
| Technologie | Waarom | Commentaar |
| NodeJS | Verplicht | / |
| Express | Vaak gebruikt in de lessen BE | Framework |
| MongoDB | Meest voor de hand liggende keuze | NoSQL database |
| Mongoose | Simpel in gebruik | Object modeling |
| Passport | Snel opzetten | Authenticatie |
| Socket-IO | Vaak gebruikt in de lessen BE | Sockets |
| Heroku | Geen Azure subscription | Hosting |

# Planning

1. Opzetten structuur zoals gezien in de lessen
2. Modellen opzetten met mongoose (users, tickets, …)
3. Users integreren met passport (authenticatie)
4. Extra logica voor events en tickets
5. Chatbox voor ieder event (socket.io)
6. Trades, wegschrijven van verruilde tickets

# Realisatie volgens planning

1. Jim Butseraen – 11/12/2015
2. Jim Butseraen & Hannes Declercq – 17/12/2015
3. Jim Butseraen – 12/12/2015
4. Hannes Declercq – 22/12/2015
5. Jim Butseraen – 27/12/2015
6. Hannes Declercq – 28/12/2015

# Toekomstplannen of mogelijke uitbreidingen

* Private chatrooms/messages

# Succesfactoren

# Conclusie: