

Konzeptbeschreibung

Team: Team 1

Mitglied 1: (Michael Hauser ,01267565)

Mitglied 2: (Angela Todhri, 11815296)

Mitglied 3: (Ismail Üner, 11721981)

Mitglied 4: (Flaminia Anselmi, 11934695)

Mitglied 5: (Sebastian Hepp, 01015083)

Mitglied 6: (Maximilian Heine, 01317323)

Proseminargruppe: Gruppe 6

Datum: 13.05.2021

Inhalt

1. Systemüberblick	3
2. Use Cases	4
2.1. Actors	4
2.2. Use-Case Diagram	5
2.3. Use-Cases	6
3. Klassendiagramm	19
4. Software-Architektur	21
4.1. Bausteinsicht	21
4.2. Verteilungssicht	22
5. GUI Prototyp	23
6. Projektplan	28

1. Systemüberblick

The TimeGuess software is a IoT- and web-based trivia game for two or more teams and is played with a TimeFlip. This is a 12-sided smart dice, whose sides are associated with different activities, points and durations.

The game is played in a web application, where the users have to login. Everything the players need to do or know can be found in the app. Before a game a user can form teams inside the app, select a topic and once everything is set up, start the game.

Once a game has started the web-application guides the teams through the game. A player has to roll the TimeFlip and the player in turn gets a new term shown on the device of another team. The player then has to make the teammates guess the term using only the in the current task stated means – pantomime, rhyming, drawing or speaking without using the term. A timer counts down the allowed time for this task.

An opponent teams must state in the app, whether the term was guessed correctly, was not guessed in time or rules were broken. Depending on this input the guessing team receives a certain amount of points.

In the game app, users can see statistics of past games and their account. Privileged users – game managers and admins – can create and delete terms and topics, specify Raspberries associated with the game and setup and create new games. Furthermore, admins can create, edit and delete users.

2. Use Cases

2.1. Actors

User

The user has an account to use the web application. All users can do actions such as login, logout, see the lobby with statistics, see their profile and join games. Player, game manager and administrator are all users with different roles.

Player

The player is a user with basic functionalities. A player is a regular player in a game without any further privileges.

Game Manager

In addition to the options of a regular user, the game manager can create and delete terms and topics, specify Raspberries associated with the game and setup and create new games. Basically, he should be the person that owns the game and sets it up for the others.

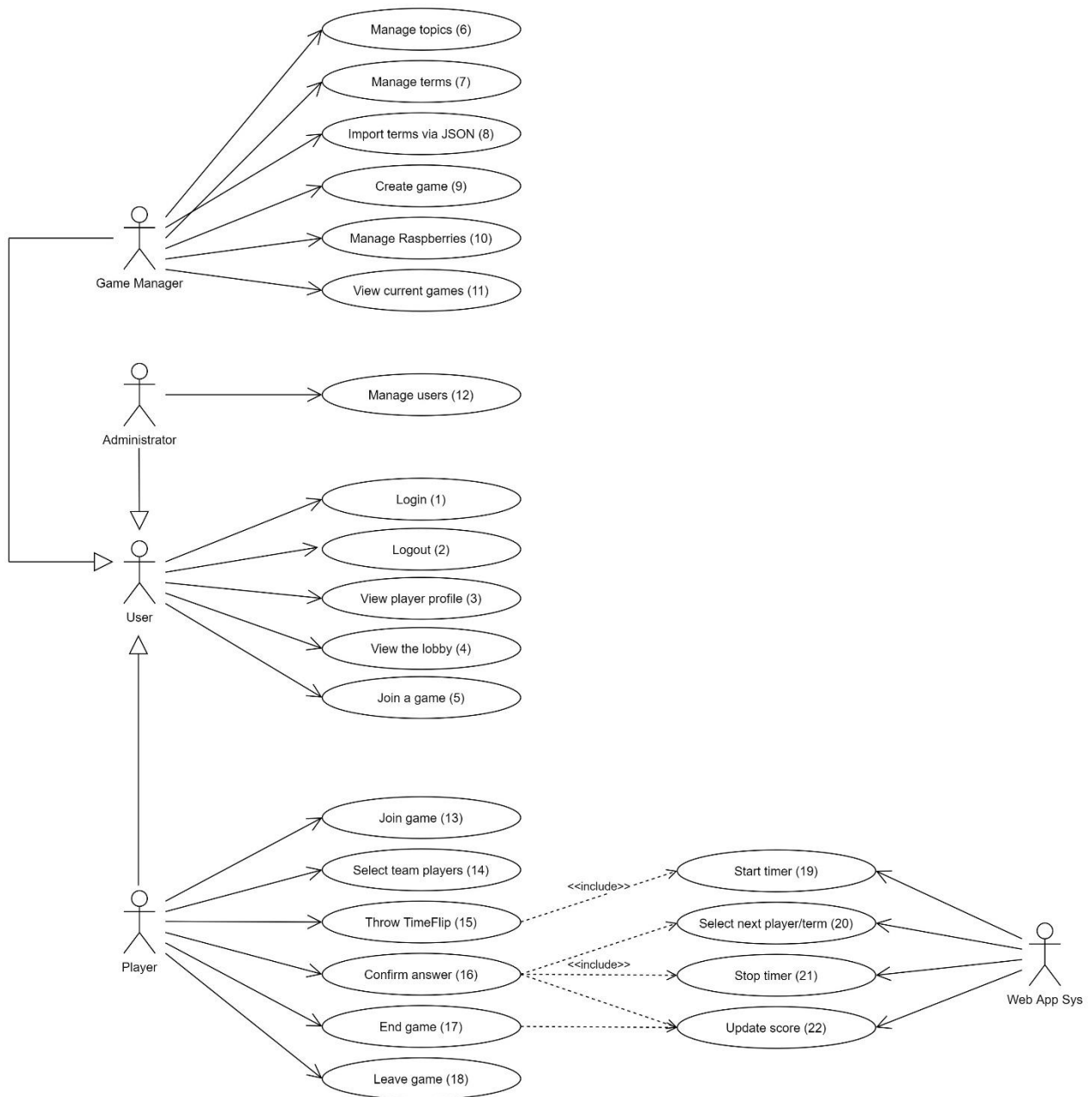
Admin

In addition to the options of a game manager, the administrator is responsible for the administration and management of players. The admin can create, delete and give rights to other users.

Web Application System

The system of the web application is responsible to randomly choose the next players and terms in the game and other tasks such as starting/ending the timer.

2.2. Use-Case Diagram



2.3. Use-Cases

2.3.1 Actor: User

Log-in (1)

Precondition:

- The system is running.
- User is on the start page.
- User exists in the database.

Procedure:

1. User enters his/her username and password
2. Clicks on "Login".

Success: User can now view the virtual game lobby.

No success: User receives an error message and is not redirected.

Log-out (2)

Precondition:

- User is logged in.

Procedure: User clicks "Log out".

Success: User is logged out and redirected to the log-in page.

View player profile (3)

Precondition:

- User is logged in.

Procedure:

1. User clicks on "My Profile".

Success: The user's profile can be viewed, along with personal information:

- Highest scored games
- Recent games
- Won games grouped by topic
- Recent teams and the teammates

View player profile: Change Raspberry (3.1)

Precondition:

- User is logged in.
- User is in the “My Profile” area.

Procedure:

1. User clicks on “Change” below “Raspberry”.
2. All Raspberries are displayed and the user can select with which he wants to associate, to be connected to its games.

Success: The selected Raspberry is set in the user’s profile.

View Lobby (4)

Precondition:

- User is logged in.

Procedure:

1. User clicks on the menu item “Lobby”.

Success: User sees global statistics and the chat.

2.3.2 Actor: Game Manager

Create topic (6.1)

Precondition:

- User is logged in.
- User has the role “Game Manager or Admin”.
- Topic is not yet in the list of topics.

Procedure:

1. User selects “Topic Hub”.
2. User selects “Create new Topic”.
3. User enters the name of the topic.
4. User selects “Save”.
 - a. User selects “Abort”: Action is aborted.

Success: The topic is added to the list of topics.

No success:

- If topic already exists, an error message is shown.
- If no name was entered, an error message is shown.

Delete topic (6.2)

Precondition:

- User is logged in.
- User has the role "Game Manager or Admin".
- Topic contains no terms.

Procedure:

1. User selects "*Topic Hub*".
2. User selects "*Delete*"-button beside the topic.

Success: The topic is removed from the list of topics.

No success: If topic contains terms, an error message is shown.

Create term (7.1)

Precondition:

- User is logged in.
- User has the role "Game Manager or Admin".
- Term is not yet in the list of terms.

Procedure:

1. User selects "*Topic Hub*".
2. User selects the "*create*"-button beside the topic he wants to add a term to.
3. User selects "*Create new Term*".
4. User enters the name of the term.
5. User selects "*Save*".
 - a. User selects "*Abort*": Action is aborted.

Success: The term is added to the list of terms.

No success:

- If term already exists, an error message is shown.
- If no name was entered, an error message is shown.

Delete term (7.2)

Precondition:

- User is logged in.
- User has the role "Game Manager or Admin".
- Topic contains no terms.

Procedure:

1. User selects "*Topic Hub*".

2. User selects the *“create”*-button beside the topic whose terms are to be edited.
3. User selects *“delete”*-button besides the term to delete.

Success: The term is removed from the list of topics.

Import terms via JSON-file (8)

Precondition:

- User is logged in.
- User has the role *“Game Manager or Admin”*.
- Topic is not yet in the list of topics.

Procedure:

1. User selects *“Topic Hub”*.
2. User selects *“Choose”*.
3. User selects the JSON-file from the devices filesystem.
4. User selects *“Submit”*.

Success:

The topic in the file is created if not existent. The terms are added to the topic.
A notification is displayed.

No success: An error message is shown.

Create game (9.1)

Precondition:

- User is logged in.
- User has the role *“Game Manager or Admin”*.

Procedure:

1. User selects *“Available Games”*.
2. User selects *“Create new Game”*.
3. User enters the name of the game.
4. User enters the points needed to win the game.
5. User selects the topic, number of teams and number of players per team.
6. User selects *“Save”*.
 - a. User selects *“Abort”*: Action is aborted.

Success: The game is added to the list of games.

No success:

- If the name already exists, an error message is shown.
- If no name was entered, an error message is shown.

- If a score for win under 12 points is entered, an error message is shown.
- If a topic is selected, that contains less than 10 terms, an error message is shown.

Add team to game (9.2)

Precondition:

- User is logged in.
- User has the role "Game Manager or Admin".

Procedure:

1. User selects "*Available Games*".
2. User selects the "*Teams*"-button next to the game to edit.
3. User selects "*Create new Team*".
4. User enters the name of the team.
5. User selects "*Save*".
 - a. User selects "*Abort*": Action is aborted.

Success: The team is added to the list of teams of this game.

No success:

- If the team name already exists in the game, an error message is shown.
- If no name was entered, an error message is shown.
- If all teams have already been created, an error message is shown.

Delete team from game (9.3)

Precondition:

- User is logged in.
- User has the role "Game Manager or Admin".

Procedure:

1. User selects "*Available Games*".
2. User selects the "*Teams*"-button next to the game to edit.
3. User selects "*Delete*"-button next to the team to delete.

Success: The team is deleted from the list of teams of this game.

Add player to team (9.4)

Precondition:

- User is logged in.
- User has the role "Game Manager or Admin".

Procedure:

1. User selects *"Available Games"*.
2. User selects the *"Teams"*-button next to the game to edit.
3. User selects *"Player"*-button next to the team to delete.
4. User selects *"Add Player"*.
5. User selects the player.
6. User selects *"Save"*.
 - a. User selects *"Abort"*: Action is aborted.

Success: The player is added to the list of players of this team.

No success:

If no free spots in the team are available anymore, an error message is shown.

Remove player from team (9.5)

Precondition:

- User is logged in.
- User has the role *"Game Manager or Admin"*.

Procedure:

1. User selects *"Available Games"*.
2. User selects the *"Teams"*-button next to the game to edit.
3. User selects *"Player"*-button next to the team to delete.
4. User selects *"Delete"*-button next to the player to remove.

Success: The player is removed from the list of players of this team.

Start game (9.6)

Precondition:

- User is logged in.
- User has the role *"Game Manager or Admin"*.

Procedure:

1. User selects *"Available Games"*.
2. User selects the *"Start"*-button next to the game to start.

Success: The user is redirected to the select-team-players-view.

No success:

- If the game was already started before, an error message is shown.
- If all teams in the game are fully set with players, but the game creator is not among those players, an error message is shown.

Edit Raspberry (10.1)

Precondition:

- User is logged in.
- User has the role "Game Manager or Admin".

Procedure:

1. User selects "*Raspberries*".
2. User selects the "*Edit*"-action next to the Raspberry to edit.
3. User can change the device name, API-key, IP-address and enable/disable it.
4. User selects "*Save*".
 - a. User selects "*Abort*": Action is aborted.

Success: The Raspberry is edited.

Deactivate Raspberry (10.2)

Precondition:

- User is logged in.
- User has the role "Game Manager or Admin".

Procedure:

1. User selects "*Raspberries*".
2. User selects the "*Invalidate API-key*"-action next to the Raspberry to disable.

Success: The API-key is removed from the Raspberry entry.

Delete Raspberry (10.3)

Precondition:

- User is logged in.
- User has the role "Game Manager or Admin".

Procedure:

1. User selects "*Raspberries*".
2. User selects the "*Delete*"-action next to the Raspberry to delete.

Success: The Raspberry is deleted from the list.

View current games (11.1)

Precondition:

- User is logged in.

- User has the role "Game Manager or Admin".

Procedure:

1. User selects "*Manager Hub*".

Success: All games are listed.

View teams of current games (11.2)

Precondition:

- User is logged in.
- User has the role "Game Manager or Admin".

Procedure:

1. User selects "*Manager Hub*".
2. User selects "*Teams*"-button next to the game.

Success: All teams of the game are listed.

2.3.3 Actor: Administrator

Create new user (12.1)

Precondition:

- User is logged in.
- User has the role "Admin".

Procedure:

1. User selects "*User Hub*".
2. User selects "*Create new User*".
3. User enters a username and password and selects the roles and the associated Raspberry.
4. User selects "*Save*".
 - a. User selects "*Abort*": Action is aborted.

Success: User is added to the list of users.

No success:

- If a field was not set, an error message is shown.
- If the username already exists, an error message is shown.

Edit user (12.2)

Precondition:

- User is logged in.

- User has the role "Admin".

Procedure:

1. User selects "*User Hub*".
2. User selects "*Edit*"-button next to the user to edit.
3. User can change the roles, the associated Raspberry and disable the selected user.
4. User selects "*Save*".
 - a. User selects "*Abort*": Action is aborted.

Success: User edited.

No success:

- If no role is set, an error message is shown.

Delete user (12.2)

Precondition:

- User is logged in.
- User has the role "Admin".

Procedure:

1. User selects "*User Hub*".
2. User selects "*Delete*"-button next to the user to edit.

Success: User is deleted.

No success:

- If user is enabled, an error message is shown.
- If user is currently logged in, an error message is shown.

2.3.4 Actor: Player

Join game (13)

Precondition:

- User is logged in.
- A game has been started by the game creator.
- The game creator is associated to the same Raspberry as the user.
- User is either already assigned to a team or there is at least one free place.

Procedure:

1. User selects "*Join Game*".

Success: User is redirected to the select-team-player-view.

No success: If a precondition is not met, an error message is shown.

Select team players (14)

Precondition:

- User is logged in.
- User has joined the game.

Procedure:

1. User selects players until all open spots in the team are taken.
2. User sets the team name or, if already set, can change it.
3. User selects *"Join Game"*.

Success:

User is displayed a message and waits until all other teams have done the same.
The user is then redirected to the game room.

No success:

- If no team name is entered, an error message is shown.
- If not all seats in the team are taken, an error message is shown.

Throw TimeFlip (15)

Precondition:

- User is logged in.
- For each team a user has joined the game and entered the game room.
- User has been chosen by the system to throw the TimeFlip.

Procedure:

1. After throwing the TimeFlip, the face at the top of the die is the one determining how and in what time the explanation has to be done.
 - There is a letter on each side:
 - P = Pantomime
 - R = Rhyme
 - S = Speaking
 - Z = Drawing
 - A number near the letter:
 - 1 = 1 point
 - 2 = points
 - 3 = 3 points
 - And another number indicating the time:
 - 1 minute
 - 2 minutes

- 3 minutes

2. The opponent's teams devices show the task for the user.

Success: The opponents teams devices show the task for the user; the timer begins.

Confirm answer (16)

Precondition:

- TimeFlip has been thrown
- Task has been displayed.
- Timer has started.
- *optional*: Timer has ended.

Procedure:

1. On opponent team selects
 - a. *"Guessed Correctly"* if the term has been guessed in time.
 - b. *"Not guessed"* if the term wasn't guessed in time.
 - c. *"Rulebreak"* if the explaining user has broken a rule.

Success: The guessing team receives:

- a. The points associated with the current side of the TimeFlip if term was guessed correctly.
- b. 0 points, if term was not guessed.
- c. -1 point if there was a break of rules.

End game (17)

Precondition:

- A team has earned at least as many points as stated for "score to win" at game creation

Success: All teams are redirected to the end view and game statistics are shown.

Leave game (18)

Precondition:

- User is in the game's end view.

Procedure:

1. User selects *"Back to Lobby"*.

Success: User is redirected to the game lobby.

2.3.5 Actor: Web Application System

Start timer (19)

Precondition:

- New round has started.
- TimeFlip die has been thrown.

Procedure: System begins the countdown, which varies depending on what side of the die is at the top.

Success: Timer starts counting and players start guessing.

Select next player/term (20)

Precondition:

- Game has started.
- Round has ended.
- Score has been updated.
- At least one player per team in virtual game room.

Procedure: System determines at random the start team and the term.

Success: The term is shown on the device of the opposing team, timer is displayed.

No success: The same term is suggested more than once in a same game.

Stop timer (21)

Precondition:

- Round has ended.
- Player has stopped the die by flipping the top side.

Procedure:

System keeps track of the time that is passing:

- Stops the countdown immediately if a player flips the time.
- Checks if the countdown has arrived to 0.

Success: Timer has stopped and now players will receive a notification to confirm their answers.

Update score (22)

Precondition:

- Round has ended.
- Players confirmed the answer.

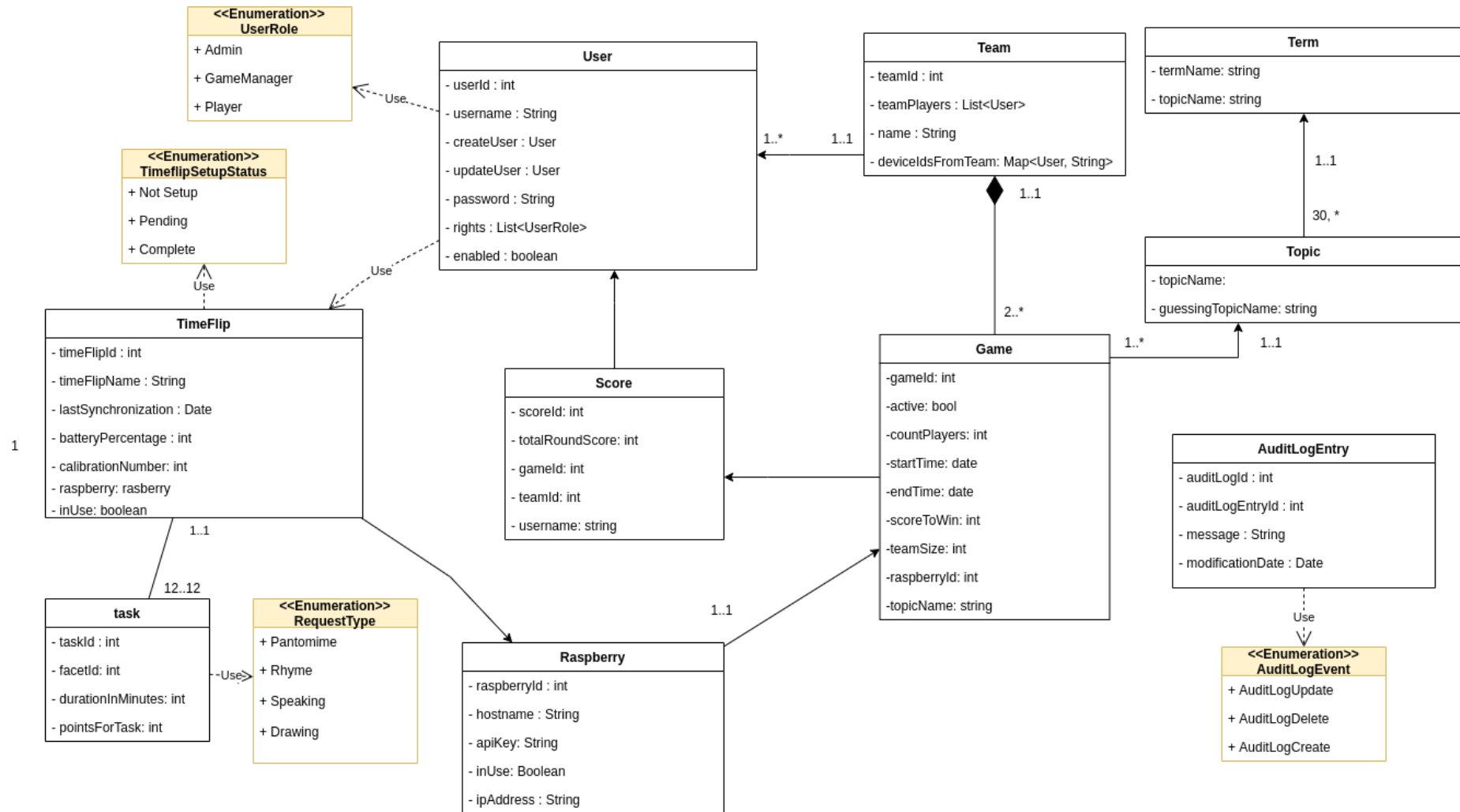
Procedure:

1. New score for the guessing team is calculated, following these options:
 - If the timer has expired completely, 0 points are awarded.
 - If the term was not guessed, 0 points are awarded.
 - If the times has not expired and the term has been guesses, the number of points written at the top side of the TimeFlip are awarded.
 - If a rule was violated, 1 point is subtracted.
2. After the score has been updated:
 - A new round is started and is time for the opposing team to play.

Success: Players start a new round.

Alternative: If the maximum of points has been reached

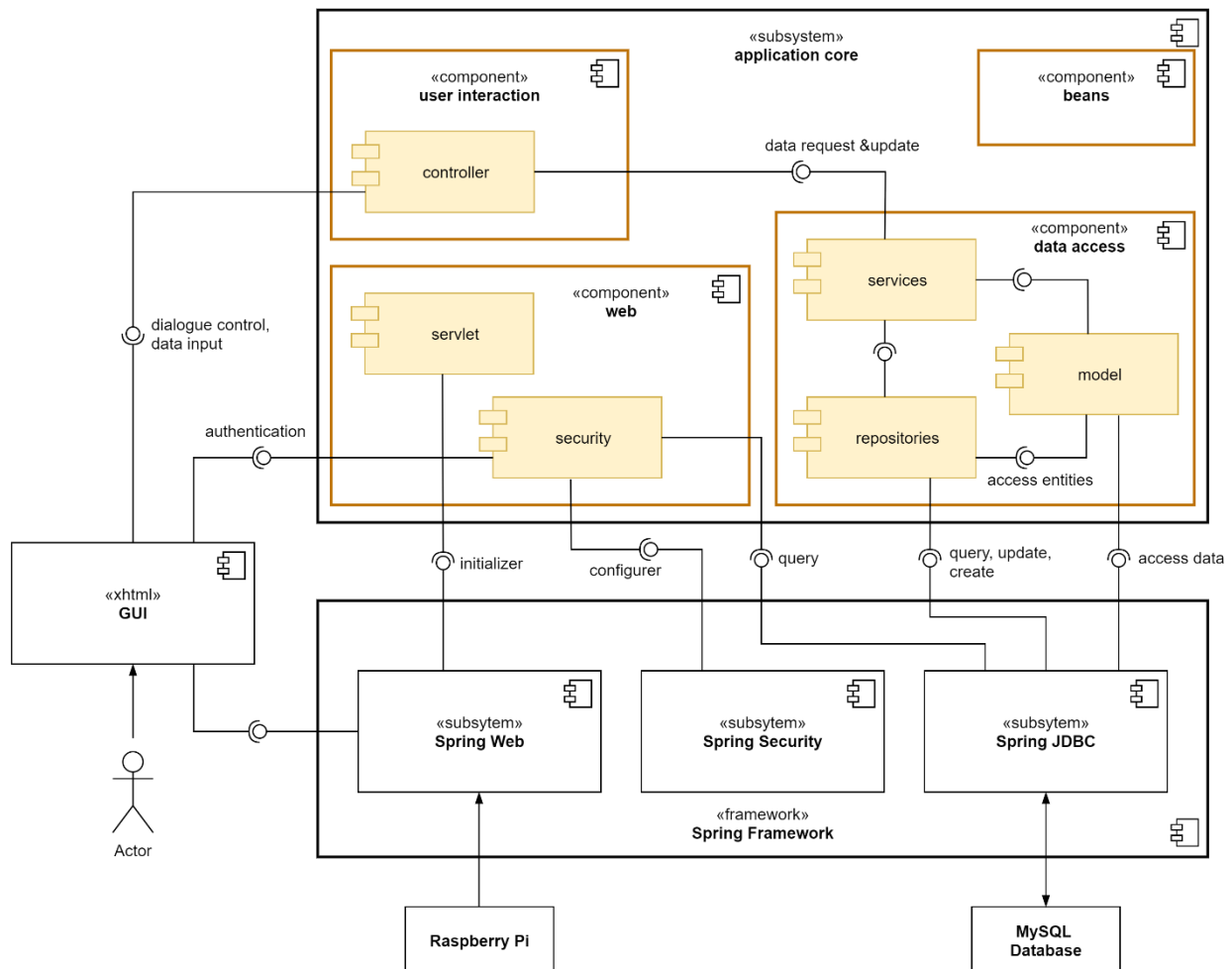
3. Klassendiagramm



class	role
Game	models a single game, with all necessary information for the gameplay; references the teams; is associated with a raspberry
Raspberry	models a single Raspberry with information necessary for the API; references a list of users, that use this device
Score	Score is instanced for each game and user and contains information for the highscore statistics; references the team the user played in
Team	models a single team in a specific game; references the users assigned to the team and their scores
Term	a single term that will be chosen from the app for players to guess
Topic	a topic will assigned to a game and references a set of terms
TimeFlipConf	contains setup-information for the TimeFlip, that specify the sides of the die
User	contains the information of a single user; references all the scores of the user, the user roles, the Raspberry the user plays on and all Teams the player was assigned to

4. Software-Architektur

4.1. Bausteinsicht

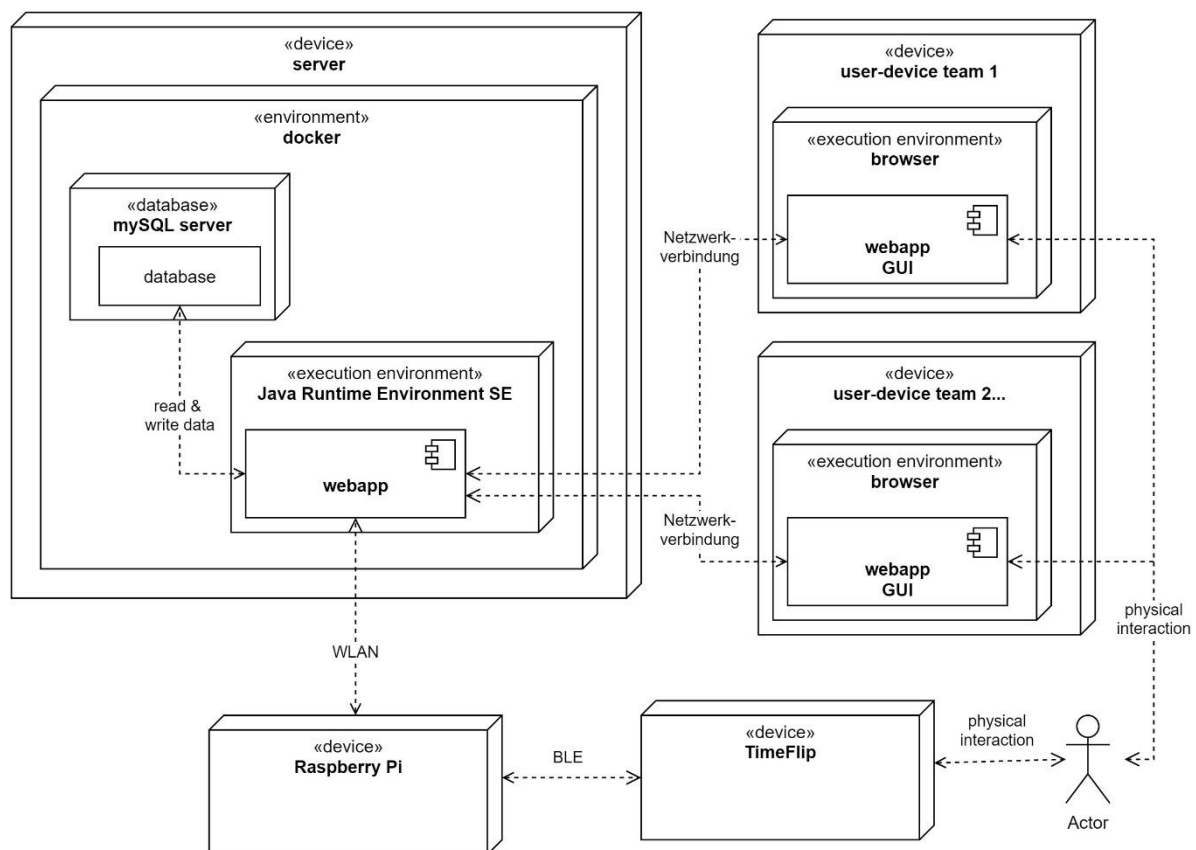


The system can be divided into the spring framework it is built upon, the graphical user interface, which is realised via XHTML, and the core of the application, which holds the actual implementation of the business logic. The application core is further broken down in the following table.

component	role
data access	retrieves, creates and updates the persisted data from the MySQL database via spring's JDBC interface
▶ model	models the persisted data as Java objects (entities)
▶ repositories	used to query, create, delete and update data in the database
▶ services	supplies methods to other components for all operations that need to access or modify persisted data, using the repositories and model modules
web	initiates and configures the web application

- **servlet** initiates the web application
- **security** manages authentication and error handling when somebody wants to access the web application
needs to query user data from the database and receives user input from the user interface
- user interaction** interface between GUI and the data model
- **controller** responsible for dialogue control of the GUI, implement the business logic initiated by the user input, retrieve the data to display from the database and modify the persisted data by communicating with the services-module
- beans** contains independently usable components that implement additional features

2.2. Verteilungssicht



5. GUI Prototyp

<Hintergrundbild>

LOGO

Log In to Your Account

☐ Keep me logged in
[Forgot Password?](#)

Log in

Need an account? [Sign up](#)

LOGIN

Dashboard
sign out

Teamname

[Dashboard](#)

angemeldet als <benutzername>

Allgemeine Statistik


Erfolgreich abgeschlossene Themengebiete:
<anzahlAbgeschlosseneThemengebiete>

Insgesamt verbrachte Zeit in Spiel:
<gesamtZeitInSpiel>

Anzahl richtig/falsch geratene Begriffe:
<anzahlRichtigeBegriffe> / <anzahlFalscheBegriffe>

Anzahl der erreichten Punkte:
<anzahlErreichtePunkte>

Ranking der Themengebiete



Themengebiet 1
 Themengebiet 2
 Themengebiet 3
Themengebiet 4
 Themengebiet 5

Vergangene Spiele

Spiel	Datum	
<spielname>	<datum>	i
<spielname>	<datum>	i
<spielname>	<datum>	i
<spielname>	<datum>	i

Meine letzten Mitspieler

Name	
<Vorname> <Nachname>	i
<Vorname> <Nachname>	i
<Vorname> <Nachname>	i
<Vorname> <Nachname>	i

Dashboard
sign out

Teamname

[Dashboard](#)

angemeldet als <benutzername>

Spielinfo zu Spiel <spielname>

Themengebiet: <themengebiet>

Gestartet: <startDate>

Teilnehmer:

Teilnehmer	Richtige/Falsche Begriffe
<Vorname1> <Nachname1>	<richtige1> / <falsche1>
<Vorname2> <Nachname2>	<richtige2> / <falsche2>
...	...

Begriffe:

Begriff	Richtig/Falsch
<begriff1>	<input type="checkbox"/>
<begriff2>	<input type="checkbox"/>

[Schließen](#)

Ranking der Themengebiete

Themengebiet 1
 Themengebiet 2
 Themengebiet 3
Themengebiet 4
 Themengebiet 5

Dashboard

sign out

Teamname

angemeldet als <benutzername>

Dashboard

Erfolgreich abgeschlossen

Insgesamt verbrachte Zeit in Räumen: <gesamteZeitInRäume>

Anzahl richtig/falsch geratene Begriffe: <anzahlRichtigeBegriffe>/<anzahlFalscheBegriffe>

Raum	Datum	
<raumid> - <raumname>	<Datum>	i
<raumid> - <raumname>	<Datum>	i
...	...	i

Letzten Spieler:

Name	
<Vorname> <Nachname>	i
...	i

Schließen

Themengebiete

Themengebiet 1

Themengebiet 2

Themengebiet 3

Themengebiet 4

Themengebiet 5

Themengebiet 1

Themengebiet 2

Themengebiet 3

Themengebiet 4

Themengebiet 5

Themengebiet 1

Themengebiet 2

Themengebiet 3

Themengebiet 4

Themengebiet 5

Lobby

sign out

Teamname

angemeldet als <benutzername>

Dashboard

Lobby

+

Neuen Raum erstellen

→

Einem Raum betreten

Lobby

sign out

Teamname

angemeldet als <benutzername>

Dashboard

Lobby

Neuen Raum erstellen

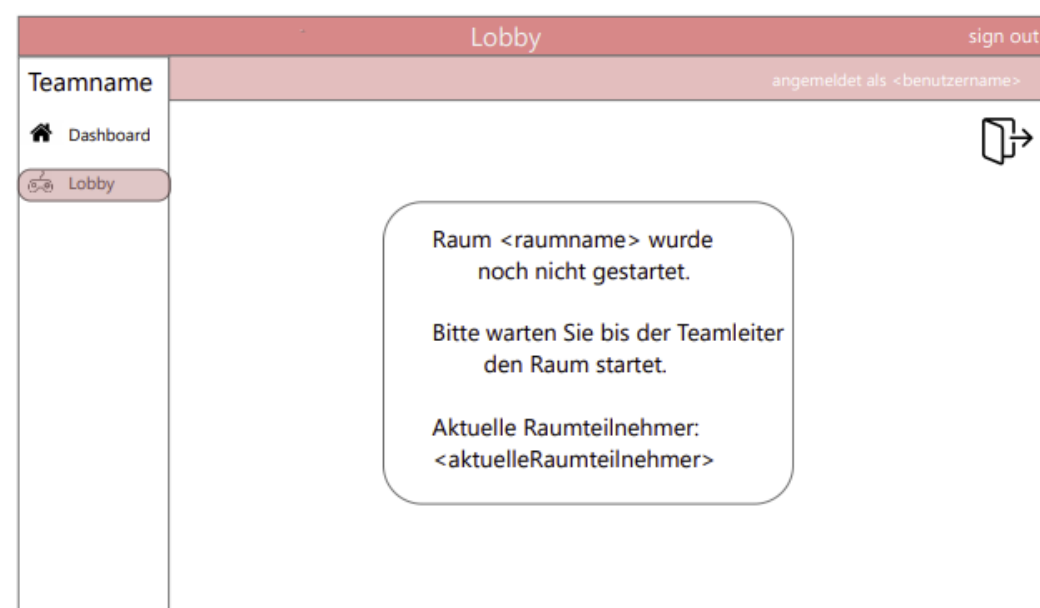
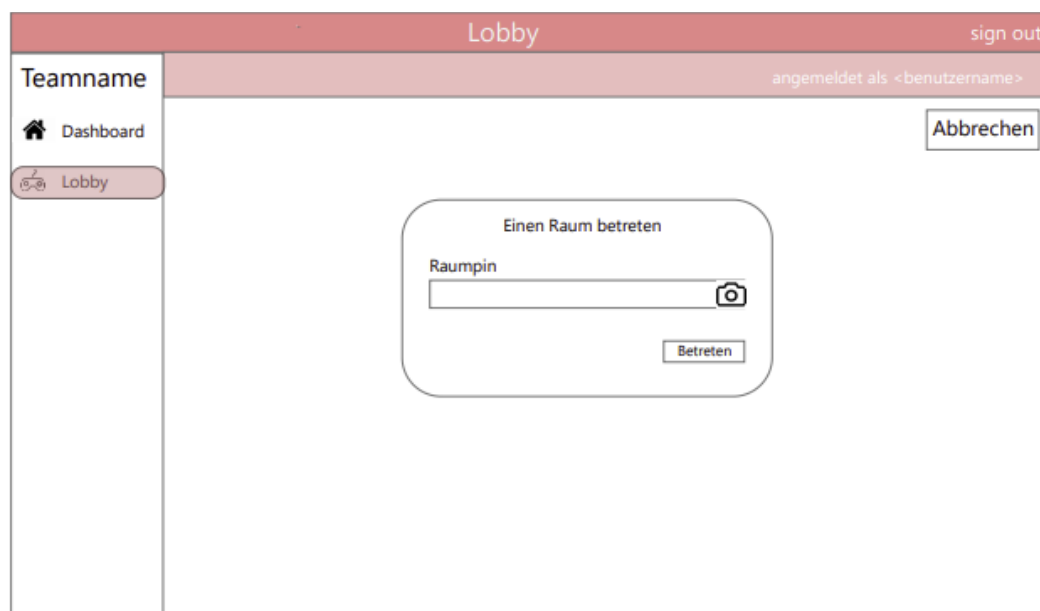
Raumname

Maximale Anzahl Spieler

Themenbereich

Punktemaximum

Erstellen



Lobby

sign out

Teamname

angemeldet als <benutzername>

Dashboard

Lobby

Zufälliges wählen des Starter Teams

<teamname>

Zufälliges wählen des Teammitglieds

<spielername>

Lobby

sign out

Teamname

angemeldet als <benutzername>

Dashboard

Lobby

Runde 12/30 Punkte: 150 15s / 30s

1. <spielername1>
2. <spielername2>
3. <spielername3>

Rateteam: <teamname>

Teammitglied: <spielername>

Würfel: <Tätigkeit>

Lobby

sign out

Teamname

angemeldet als <benutzername>

Dashboard

Lobby

Runde 12/30 Punkte: 150

Begriff: <begriff>

Bestätigung richtig Erraten: Wurde der 1. Begriff richtig erraten?

JA NEIN

Teamname

Dashboard

Lobby







Themenbereich

sign out

angemeldet als <benutzername>

Anzahl Themenbereiche: <anzahlThemenbereiche>

Hinzufügen

Themenbereich	Anzahl Begriffe	
<input type="text"/>		
Themenbereich 1	<anzahlBegriffe1>	 
Themenbereich 2	<anzahlBegriffe2>	 
Themenbereich3	<anzahlBegriffe3>	 

6. Projektplan

Nr.	Milestone	Edited by	App. Time	Deadline
	Previous tasks			
1	Konzeptbeschreibung	All	50:00	18.03.
2	Model	Flaminia/Angela	35:00	25.03.
3	Landing Page	Ismail	10:00	25.03.
4	Raspberry Pi Setup	Michael/Max	15:00	28.03.
5	TimeFlip Setup	Michael/Sebastian	15:00	03.04.
6	REST API	Max/Sebastian	30:00	18.04.
7	Database data	Angela	20:00	01.04.
8	MySQL	Flaminia	20:00	01.05.
	Recent tasks			
8	Advanced Backend Functionality Controllers, Services, Repository	All	80:00	13.04.
9	Advanced Frontend	Sebastian /Max	50:00	20.04
10	Data Management -create game statistics -create terms	Ismail/Michael	20:00	27.04
11	JSON integration	Flaminia	20:00	02.05.
12	Bugfixing I	All	50:00	09.05.
13	Testdrehbuch	Falminia / Angela	40:00	13.05.
14	Hardware integration	Michael / Max		12.05.
15	Stable and working system	All		13.05.2021
	Upcoming tasks			
16	Acceptance Tests	All		20.05.
17	JUnit Tests	Flaminia / Angela	50:00	16.06.
18	Dockerization		25:00	06.06.
19	Systemtest	All	30:00	06.06.
20	Bugfixing II	All	30:00	13.06.
21	Code doku		10:00	13.06.
22	Softwarekonzept update		20:00	13.06.
23	Abschlussbericht		10:00	13.06.
24	Final project results	All	10:00	18.06.2021
25	Final presentation	All	08:00	21.06.2021