Software Design

Ac. yr. 2019./2020.

Festival organization

Documentation, Rev. *<1>*

Group: *Festival organisation*

Coordinator: *Bartol Bilic*

Turn-in Date: *<14>*. *<11>*. *<2019>*.

Supervisor: *Hrvoje Nuic*

Contents

1 Documentation Change Log **3**

2 Project Description **4**
2.1 General Idea 4
2.2 Position in the market - the competition 4
2.3 In-Depth Description 5
2.3.1 Account Data 5
2.3.2 User roles 6
2.3.3 Administrators 6
2.3.4 Creators 6
2.3.5 Organiser 6
2.3.6 Worker 7
2.3.7 Functions and happenings of the system 7
2.3.8 Festival 8
2.3.9 Auction 8
2.3.10 Timelines 8
2.3.11 Job 9
2.3.12 Activity 9
2.3.13 Cards 9
2.3.14 Card Format 9
2.3.15 System implementation targets 10
2.3.16 Project Scope and Targeted Users 10
2.3.17 Changes, upgrades, adaptability of the Application 10

3 Software Specification **12**
3.1 Functional Requirements 12
3.1.1 Use Cases 15
3.1.2 Sekvencijski dijagrami 43
3.2 Ostali zahtjevi 44

4 Architecture and System Design **46**
4.1 Database 46
4.1.1 Tables details 47
4.1.2 Database diagrams 50
4.2 Class Diagram 52
4.3 Dijagram stanja 53
4.4 Dijagram aktivnosti 54
4.5 Dijagram komponenti 55

5 Implementation and User Interface **56**
5.1 Korištene tehnologije i alati 56
5.2 Ispitivanje programskog rješenja 57
5.2.1 Ispitivanje komponenti 57
5.2.2 Ispitivanje sustava 57
5.3 Dijagram razmještaja 58
5.4 Upute za puštanje u pogon 59

6 Conclusion and Outline of Planned Future Work **60**

Popis literature **61**

Image and diagram index **62**

Appendix: Preview of group activity **63**

1. Documentation Change Log

Rev.	Opis promjene/dodatka	Authors	Date
0.1	Template uploaded to git.	Bilic	20.10.2019.
0.2	Project Description written.	Ceple	6.11.2019.
0.3	Functional Requirements written.	Ceple	6.11.2019.
0.4	Class Diagram	Strbad	12.11.2019.
0.5	Use Cases finished	Ceple	13.11.2019.
0.6	Database description and diagrams added	Fribert	13.11.2019.

Moraju postojati glavne revizije dokumenata 1.0 i 2.0 na kraju prvog i drugog ciklusa. Između tih revizija mogu postojati manje revizije već prema tome kako se dokument bude nadopunjavao. Očekuje se da nakon svake značajnije promjene (dodatka, izmjene, uklanjanja dijelova teksta i popratnih grafičkih sadržaja) dokumenta se to zabilježi kao revizija. Npr., revizije unutar prvog ciklusa će imati oznake 0.1, 0.2, . . . , 0.9, 0.10, 0.11.. sve do konačne revizije prvog ciklusa 1.0. U drugom ciklusu se nastavlja s revizijama 1.1, 1.2, itd.

2. Project Description

2.1 General Idea

The idea of this app is to enable a low to mid size festival organisation in a relatively simple and straight-forward manner that would be easily accessible and understandable even to non-technically educated Users.

The application would be open-source, and would run on the Android OS - a native mobile app.

2.2 Position in the market - the competition

Mainly, the competition consists of either high-profile professional apps, or non-native(non-mobile) apps. Thus, the point of this app is to fill that gap - it's supposed to be a native app, that's relatively simple to use, and very portable and easily deployable.

This would also imply, and goes hand in hand, with the fact that the application would be easy to use and easily accessible to a wide range of people - from highly-trained professional IT Users all the way to non-IT savvy amateur/inexperienced Users.

Because of the low difficulty, and relatively ad rem employability, this app would be more suitable to the lower skill level(entry to mid-level) Users, as it is likely that Pro Users would require a larger scale App for, probably, larger caliber Festivals that they deal with. With that in mind however, this app can be used as a mini, mobile device reminder version of whatever Pro tool is used for Festival organisation.

In the market there is presence of both event organisation, and user-event interface apps. This app is focused on organisation, and not festival-goers. Therefore, from a marketplace standpoint, there will be no impact on the demand of the application due to the selected specialisation.

2.3 In-Depth Description

Application would be used for Festival organisation - music festivals, film festivals, library events, food and alcohol festivals, parties, birthdays, and other kinds of meet-ups. By scope and complexity it would be used for smaller and medium scale Events/Festivals.

The system would be run and moderated by Administrators. The system/platform would allow concurrent usage by multiple Users. These Users are: Administrators, Creators, Organisers, Workers.

From here on, unregistered users and Users who aren't logged in will be referred to as Guests.

2.3.1 Account Data

Prior to registration, Guests must fill in a Registration form. This form consists of:

- Username
- Email
- Password
- Password Verification
- Phone Number
- Name
- Surname
- Desired Role
    1. Leader
    2. Organiser
    3. Worker

Aside from this data, Users can also upload their Profile Picture. In case they don't, a default anonymous one will be used. These Profile Pictures will be put on Users' Cards that serve as Festival entrance tickets.

Users can change some of this data. Specifically, they can alter:

- Email

- Password
- Phone Number
- Profile Picture

Users aren't allowed to modify their Usernames in order to prevent abuse and misuse. If Users want their Usernames, Names or Surnames changed, they can contact the Administrator, and if the appeal makes sense the Administrator can change those values for the User.

2.3.2 User roles

2.3.3 Administrators

Maintain and organize the platform. They curate the Creators. They have complete transparency and access to all data. They can veto and issue bans on: Festivals, Jobs, User registrations, comments, etc... Basically they have complete control over the platform.

2.3.4 Creators

Need to be verified by one of the Administrators. Have the ability to create and edit Festivals, as well as have complete transparency of the Festivals they have created. Can undertake certain actions regarding the details of their Festivals(and Jobs).

They appoint and verify Organisers. To these Organisers they assign Festivals that need to be organised - delegation of Festival planning and execution. Organisers cannot organise concurrent Festivals.

2.3.5 Organiser

Organisers are appointed by the Creators. They manage, plan and execute Festivals on both a macro and micro/detail scale. They are appointed to Festivals by Creators.

They can organise multiple Festivals, and be appointed to those Festivals by multiple Creators - it is only important that none of those Festivals are concurrent.

They organise and manage Jobs and Activities that need to be done for the Festival. Jobs are performed by Workers. Organisers also have the ability to write and read comments on the Worker's walls as to better organise these Jobs.

Job and Activity management is done via Auctions. First Workers apply to these Auctions, and their entries need to be verified by the Organisers. And then the Verified Entries can compete to receive the Job Offer. Should they deem it necessary, Organisers can extend Auctions by one day.

2.3.6 Worker

They perform Jobs and Activities. For these Jobs they first have to apply to the selection process(Auctions) during which their entries are curated by the Organisers(or eventually Administrators or Creators). Upon verification, their entries compete in order for the Worker to receive the Job Offer.

Some Jobs are necessary to be done by multiple Workers. Workers can specify in their Auction Entry how many more Workers would the Job require. Jobs can be done concurrently(parallelised), and Workers can perform Jobs with maximum freedom, so long as these Jobs aren't concurrent - in case they are, the system will immediately veto such entries.

Workers' accounts have certain specifics compared to other accounts. Their profiles contain:

1. Field of specialisation
2. Basic account data and information
3. Former Job information
4. Comment section on the Worker's profile wall – intended to allow the Worker's co-workers, boss, ... to comment the Worker's performance, characteristics, their satisfaction of working with the Worker, etc...

2.3.7 Functions and happenings of the system

Here some inner elements of the system will be defined, along with their attributes and characteristics.

2.3.8 Festival

Festivals are events that are being organised. Their attributes are:

1. Name
2. Description
3. Location
4. Planned start-end time

2.3.9 Auction

Auctions are part of the process where Workers apply to Jobs. First Workers need to turn in their entries, which are then verified by the Organisers/Creators, and moderated by Administrators. Upon successful verification/moderation, Workers' entries finally enter the Auction competition where the lowest bid ends upon the expiration of the Auction period. If need be, Organisers and Creators can extend the Auction period by one day.

Auctions have the following attributes:

1. Price
2. Comment
3. Number of Workers needed(Workforce Quantity)
4. Estimated time to Completion(ETC)

2.3.10 Timelines

Timelines are intended to provide an easy and graphical way of organising Jobs in the time-domain – this is done in a way that the timeline itself is actually a flowchart organised against a time axis. The liable User can add, remove and manipulate the Objects located in the Timeline.

The beginning time, end time and the duration of a Timeline Object can be manipulated by moving and resizing the Object's corresponding box. This provides a visual way of organising Objects, and makes it easy to parallelise them.

The ability to parallelise implies the existence of multiple branches. Branches can be created, deleted, and moved.

There are 2 types of Timelines:

- Festival Job Timeline
- Job Auction Timeline

2.3.11 Job

Jobs are performed by Workers. They are constituted of Activities that need to be done in order for the Job to be completed. Jobs can require multiple Workers. The lowest bid Worker is assigned the Job. Auctions usually last 1 day, but if necessary Organisers and/or Creators can extend them by an additional day.

Jobs are to be given a sequential order of execution. They can be parallelised as well. Multiple Workers cannot work on concurrent Jobs.

2.3.12 Activity

Jobs consist of multiple Activities that need to be done in order for the Job to be completed - as already defined above. Activities help break down Jobs into manageable, and easily organised chunks. Since Jobs can be done by multiple Workers, as well as be parallelised, Activities can help with that task as they would be the elementary particle, the smallest unit of work that needs to be done and distributed throughout the network of Workers that would be performing the given Job.

2.3.13 Cards

*Participants in the organisation of the Festival would each receive a **SINGLE** card - with an exception of multiple Workers working on the same Job - that will be explained further below. Workers also have another specificity - their Cards feature some additional information.*

A single Card can be printed for a single User and a single Festival. Therefore, the same User is able to download and print multiple Cards for multiple Festivals.

2.3.14 Card Format

The cards can be downloaded in a .pdf format - with the dimensions 10x7cm

Everyone's Cards feature the following information:

1. User picture

2. Name and Surname
3. Name and Logo of the Festival
4. QR code(MD5 Hash):
   (a) User's Name
   (b) Name of the Festival

*Workers' Cards contain **ADDITIONAL** information:*

1. Time and Location of the Job
2. QR code(MD5 Hash):
   (a) ID of the Job
   (b) ID of the User

In case a Job needs multiple Workers - multiple Cards will be printed for all the Workers.

2.3.15 System implementation targets

*The Platform is meant to enable **MULTIPLE** Users **CONCURRENT** access to, and usage of the Platform. This would make festival organising a dynamic and fast environment. As such, a modern object-oriented programming language will be used for implementation.*

Since the application will be developed for Android OS, we have chosen Java as the programming language, and SQLite has been chosen for the database software.

2.3.16 Project Scope and Targeted Users

The software would be targeted towards people organising a festival and/or participating in its execution - administrators, technicians, investors, musicians, other kinds of artists, influencers, ...

The software, as said before, is aimed at primarily festivals of smaller sizes, but could be employed for festivals up to certain 'medium' size. Especially with improvements, it could be a viable free, open-source and easily protable alternative for both medium-sized festivals and small-sized festivals.

2.3.17 Changes, upgrades, adaptability of the Application

The application as it stands currently has some unfortunate restrictions on the freedom that Administrators, Creators and Organisers. It would be possible to modify these

restrictions as to allow a greater freedom of Festival organisation, but still keeping some in place as to prevent abuse and misuse. Certain functionalities could also be added in order to make the app more useful to mid-sized Festivals.

A few changes that would probably help tremendously is to implement an intra-platform messaging service, feedback system, and support/ticked service. This would allow the communication within the App between Users. It would also provide a way for the Developers to easily diagnose, track and reproduce bugs, as well as see what changes, ideas and updates Users would like to see in the App. Finally, a support/ticket system would help alleviate any frustration that Users could suffer due to bugs and/or failures of the platform.

A list of changes(actively tracked) to eventually, if time and will permits, be implemented:

- Email confirmation
- Email reset

3. Software Specification

3.1 Functional Requirements

Stakeholders:

1. Developers and Maintainers
2. Festival-goers
3. Festival investors and sponsors
4. Administrator
5. Creator
6. Organiser
7. Worker

Actors and their functional requirements:

1. Unregistered/Guest User(initiator) can:
   (a) Register a new account - fill in the form
      i. Username
      ii. Email
      iii. Password
      iv. Password Verification
      v. Phone Number
      vi. Name
      vii. Surname
      viii. Desired Role
         A. Leader
         B. Organiser
         C. Worker
   (b) Log in - fill in the form
      i. Username or Email
      ii. Password

2. Administrator can:
   (a) Access the list of Users
   (b) Verify Creators
   (c) Moderate Users' details and/or ban them as to alleviate abuse/misuse
   (d) Access the list of Festivals
      i. Access the list of Jobs
         A. Access the list of corresponding Activities
         B. Access the list of Workers
      ii. Moderate Festivals and/or veto them as to alleviate abuse/misuse
3. Leaders manage Festivals:
   (a) Create [multiple] Festivals
   (b) Modify or delete their Festivals
   (c) **Inherit Organiser functionalities for their Festivals**
   (d) Appoint Organisers to their Festivals(check if the selected Organiser is organising any possibly concurrent Festivals)
   (e) Access the Jobs, Activities, Workers and other details of their Festival
4. Organiser organises the concrete Festival workflow:
   (a) Job management
      i. Select which Jobs need to be done - open their corresponding Job Auctions
      ii. Job Sequence - order and parallelise Jobs
      iii. Ability to extend Job Auction lifetime by 1 day
      iv. View and modify Jobs
         A. Access Workers' profiles, details, comments, ...
         B. Access Job description, Time and Location - modify them as needed
         C. View each Job's list of Activities
   (b) Organise a Festival – Ability to organise multiple Festivals - check for concurrency!
5. Workers perform specific Jobs. If necessary, multiple Workers work on the same Job. They can
   (a) Select their fields of specialisation
   (b) Apply to Job Auctions
   (c) Perform Job - can perform multiple Jobs - check for concurrency!

(d) Fill out Job information sheet

i. Job Description

ii. Job Location and Time

iii. Form a list of Activities that need to be done - ability to modify, add and/or delete the entries in this list

3.1.1 Use Cases

dio 1. revizije

Use Cases Description

Funkcionalne zahtjeve razraditi u obliku obrazaca uporabe. Svaki obrazac je potrebno razraditi prema donjem predlošku. Ukoliko u nekom koraku može doći do odstupanja, potrebno je to odstupanje opisati i po mogućnosti ponuditi rješenje kojim bi se tijek obrasca vratio na osnovni tijek.

UC<broj obrasca> -<ime obrasca>

- **Glavni sudionik:** <sudionik>
- **Cilj:** <cilj>
- **Sudionici:** <sudionici>
- **Preduvjet:** <preduvjet>
- **Opis osnovnog tijeka:**
  1. <opis korak jedan>
  2. <opis korak dva>
  3. <opis korak tri>
  4. <opis korak četiri>
  5. <opis korak pet>
- **Opis mogućih odstupanja:**

  2.a <opis mogućeg scenarija odstupanja u koraku 2>
    1. <opis rješenja mogućeg scenarija korak 1>
    2. <opis rješenja mogućeg scenarija korak 2>

  2.b <opis mogućeg scenarija odstupanja u koraku 2>

  3.a <opis mogućeg scenarija odstupanja u koraku 3>

UC1 - Festivals Overview and Detail inspection

- **Main Stakeholders:** Administrators
- **Goal:** View the list of all the Festivals, ability to click on a specific Festival and view its details in a new Screen
- **Stakeholders:** Database
- **Conditions:** -
- **Event flow description:**

1. A list of Festivals is displayed(retrieved from the Database)
2. Administrator selects a Festival of which he wants to inspect further details
3. A new Screen appears depicting a detailed view of the Festival's information

UC2 - Registration

- **Main Stakeholders:** Unregistered/Guest Users
- **Goal:** Register a new User Account
- **Stakeholders:** Database
- **Conditions:** Must not be logged in, and must be located at the Login screen.
- **Event flow description:**

1. User is located at the login screen, and taps 'Create one'(located next to 'No account yet?')
2. A new Screen pops up - Guest fills the Registration Form
3. Upon tapping 'Create Account' a new account is created and stored in the Database

UC3 - Log-In

- **Main Stakeholders:** Guest Users who have a registered account
- **Goal:** Log into the platform
- **Stakeholders:** Database
- **Conditions:** Must not be logged in, and must be located at the Login screen.
- **Event flow description:**

1. User enter the email or username and their password
2. User taps the LOGIN button
3. Database checks the data and if login is successful the user is logged in

3.a Email/Username and Password combination is wrong and the user isn't logged in.

1. The fields Email/Username and Password are reset
2. An error message is displayed
3. User needs to login again

UC4 - Account Data Overview

- **Main Stakeholders:** User

- **Goal:** View the account data
- **Stakeholders:** Database
- **Conditions:** Must be logged in
- **Event flow description:**

1. User taps the sandwich button in the upper right screen corner, and then taps Account
2. Data is retrieved from the Database
3. A screen depicting Account data appears

2.a Data not successfully retrieved due to connection timeout(can be caused by various network problems)

1. An error message is displayed informing the User data couldn't be retrieved
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

UC5 - Changing Account Data

- **Main Stakeholders:** User
- **Goal:** Change the account data
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in
- **Event flow description:**

1. User taps the sandwich button in the upper right screen corner, and then taps Account
2. Data is retrieved from the Database
3. A screen depicting Account data appears
4. User clicks the 'Change info' button and is taken to a new Screen
5. User enters the new data desired
6. User clicks the 'OK' button and the data is sent to the Database
7. Upon receiving a confirmation from the Server that change has taken place the notification is displayed to the User

2.a Data not successfully retrieved due to connection timeout(can be caused by various network problems)

1. An error message is displayed informing the User data couldn't be retrieved
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

6.a Data not successfully sent due to connection timeout(can be caused by various network problems)

1. Field values are saved, and kept the same
2. An error is displayed to the User stating that data wasn't successfully sent and that User needs to resend the form
3. User resends the form. If still not successful, the same steps above apply.

7.a An error on the Server occurs and account data isn't updated properly.

1. Field values are saved, and kept the same
2. An error is displayed to the User stating that an error has occurred and that User needs to resend the form
3. User resends the form. If still not successful, the same steps above apply.

7.b User enters disallowed/illegal/rubbish data and the Server doesn't accept such an attempt

1. The wrong field values are reset
2. The User is notified that they have entered improper input. The notification mentions the limits and asks the User to change the invalid fields
3. The User enters the data again. If the input is illegal the same steps above apply.
4. Data is accepted and saved this time

7.c No confirmation from the Server(probably a network error)

1. Field values are saved, and kept the same
2. An error is displayed to the User stating that a confirmation wasn't received and that User needs to resend the form
3. User resends the form. If still not successful, the same steps above apply.

UC6 - Account Deletion

- **Main Stakeholders:** User
- **Goal:** Delete his Account
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in
- **Event flow description:**

1. User taps the sandwich button in the upper right screen corner, and then taps Account
2. Data is retrieved from the Database
3. A screen depicting Account data appears
4. User clicks the 'Delete Account' button
5. A request is sent to the Server. Upon reception, the Server deletes the Account from the database.
6. The Server sends the confirmation to the User that his account has been deleted.
7. The User is logged out of the application

2.a Data not successfully retrieved due to connection timeout(can be caused by various network problems)

1. An error message is displayed informing the User data couldn't be retrieved
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

5.a Request not successfully sent due to connection timeout(can be caused by various network problems)

1. An error is displayed to the User stating that the request wasn't successfully sent and that User needs to resend it
2. User resends the request. If still not successful, the same steps above apply.

7.a An error on the Server occurs and account isn't deleted.

1. Field values are saved, and kept the same.
2. An error is displayed to the User stating an error has occurred and that User needs to resend the form
3. User resends the form. If still not successful, the same steps above apply.

UC7 - Verifying a Leader

- **Main Stakeholders:** Administrator
- **Goal:** Check, and if all is in order, verify a Leader
- **Stakeholders:** Database, Leader
- **Conditions:** Must be logged in. There is a Leader awaiting confirmation.
- **Event flow description:**

1. Administrator opens the panel for verifying Leaders
2. Data is fetched from the Database
3. Administrator reads the info about the Leader and the Festival that the Leader wants to create
4. Administrator decides whether to verify the Leader or not

2.a Data not successfully retrieved

1. An error message is displayed informing the User data couldn't be retrieved
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

UC8 - Create a Festival

- **Main Stakeholders:** Leader
- **Goal:** Create a Festival, and open it up to Workers so that they can start working
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in
- **Event flow description:**

1. The Leader opens the Screen for creating a Festival
2. The Leader fills in all the necessary info required for creating a Festival
3. The Leader submits the form and a Festival is created

2.a Illegal input

1. An error message is displayed informing the Leader that he has enter an illegal input
2. The notification asks the Leader to re-format and re-enter the input. The expected format and rules are displayed

3. The steps above are repeated if new input isn't accepted either

3.a Data not successfully sent and parsed on the Server

1. An error message is displayed informing the User that an error has occurred
2. A 'Retry' button is displayed on the screen - upon clicking it the form is resent and parsing tried again
3. The Leader is notified upon success

UC9 - Appoint an Organiser to the selected Festival

- **Main Stakeholders:** Leader
- **Goal:** The Organiser is appointed and begins carefully managing and organising the Festival
- **Stakeholders:** Database, Back-End(Server), Organiser, Festival
- **Conditions:** Must be logged in, a Festival requires an Organiser
- **Event flow description:**

1. The Leader opens the Screen featuring their festivals
2. The Leader selects one of the Festivals
3. The Leader selects one of the Organisers, and appoints him to the selected Festival
4. This data is sent to the Server, which updates the Database
5. A notification is sent to the Organiser, who can either accept or reject the said Festival

4.a Data not successfully sent and/or parsed on the Server

1. An error message is displayed informing the User that an error has occurred
2. The Leader can try resending the form
3. The Leader is notified upon success

5.a Attempt to send a notification to the Organiser fails.

1. The Server will resend it until success
2. Upon success, the Leader is notified that the action of requesting an Organiser is successful

UC10 - Update Festival info

- **Main Stakeholders:** Leader
- **Goal:** Create a Festival, and open it up to Workers so that they can start working

- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in
- **Event flow description:**

  1. The Leader opens the Screen for creating a Festival
  2. The Leader fills in all the necessary info required for creating a Festival
  3. The Leader submits the form and a Festival is created

  2.a Illegal input
    1. An error message is displayed informing the Leader that he has enter an illegal input
    2. The notification asks the Leader to re-format and re-enter the input. The expected format and rules are displayed
    3. The steps above are repeated if new input isn't accepted either

  3.a Data not successfully sent and parsed on the Server
    1. An error message is displayed informing the User that an error has occurred
    2. A 'Retry' button is displayed on the screen - upon clicking it the form is resent and parsing tried again
    3. The Leader is notified upon success

UC11 - Create a Job

- **Main Stakeholders:** Organiser
- **Goal:** Create a Job entry that would be visible to Workers who would apply to this Job
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in
- **Event flow description:**

  1. The Organiser opens the Screen for creating a Job
  2. The Leader fills in all the necessary info required for creating a Job
  3. The Leader submits the form, the Job is created and added to the list of Jobs(this list is visible to Workers who can then send their application to Organisers for the selected Job)

  2.a Illegal input
    1. An error message is displayed informing the Organiser that they have entered an illegal input

2. The notification asks the Organiser to re-format and re-enter the input. The expected format and rules are displayed
3. The steps above are repeated if new input isn't accepted either

3.a Data not successfully sent and parsed on the Server

1. An error message is displayed informing the User that an error has occurred
2. A 'Retry' button is displayed on the screen - upon clicking it the form is resent and parsing tried again
3. The Organiser is notified upon success

UC12 - View the list of all the Jobs(Worker View)

- **Main Stakeholders:** Worker, Administrator
- **Goal:** View the list of Jobs. Includes the ability to filter the Jobs by Categories.
- **Stakeholders:** Database
- **Conditions:** Must be logged in
- **Event flow description:**

1. The User opens the Screen for viewing the list of Jobs
2. The data is fetched from the Database
3. Optional: Filtering the Jobs according to the specified filter
4. The User selects the Job and views the details about it(the 'View Job details' button)

2.a Illegal input

1. An error message is displayed informing the Organiser that they have entered an illegal input
2. The notification asks the Organiser to re-format and re-enter the input. The expected format and rules are displayed
3. The steps above are repeated if new input isn't accepted either

UC13 - View Job details(Worker view)

- **Main Stakeholders:** Worker, Administrator
- **Goal:** View Job details and specifics - such as Time, Location, Duration, Description, ...
- **Stakeholders:** Database
- **Conditions:** Must be logged in
- **Event flow description:**

1. The User can read Job specifics
2. The Worker can click the 'Send Job application' button in order to apply to the given Job
3. The Administrator can click the 'Moderate this Job' button

1.a Data not successfully retrieved
   1. An error message is displayed informing the User data couldn't be retrieved
   2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
   3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

UC14 - Apply to the Job(Worker)

- **Main Stakeholders:** Worker
- **Goal:** Send his application to the Job's supervising Organiser
- **Stakeholders:** Database, Back-End(Server), Organiser
- **Conditions:** Must be logged in, the Job details Screen is opened
- **Event flow description:**
   1. The Worker on this Screen fills out the details into the form(Application Form)
   2. Upon filling out the form, he sends the form by pressing the 'Send Job Application' button
   3. The Application is sent to the Server

3.a Data not successfully sent to the Server
   1. An error message is displayed informing the User data wasn't successfully sent to the Server
   2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
   3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC15 - Moderate the Job

- **Main Stakeholders:** Administrator
- **Goal:** Moderating the selected Job - in case certain part of it violates the Rules
- **Stakeholders:** Database, Back-End(Server)

- **Conditions:** Must be logged in
- **Event flow description:**

1. The Administrator alters certain part of the Job, or deletes it(by pressing the 'Delete Job' button)
2. The Administrator chooses whether the Job creator(corresponding Organiser or Leader) is notified or not
3. The changes are sent to the Server

3.a Data not successfully sent to the Server

1. An error message is displayed informing the User data wasn't successfully sent to the Server
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC16 - View the list of this Festival Jobs

- **Main Stakeholders:** Administrator, Organiser, Leader
- **Goal:** View the list of Jobs for the currently selected Festival
- **Stakeholders:** Database
- **Conditions:** Must be logged in
- **Event flow description:**

1. The data is fetched from the Database
2. The list is displayed to the User
3. The User can press the 'View Job details' button which will take them to a new Screen where they can view the Job details and specifics, modify or delete the Job

1.a Data not successfully retrieved

1. An error message is displayed informing the User data couldn't be retrieved
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

UC17 - View Job details(Organiser view)

- **Main Stakeholders:** Administrator, Leader, Organiser
- **Goal:** View the details and specifics of the selected Job. Can also proceed to the Screen for modifying or deleting the Job
- **Stakeholders:** Database
- **Conditions:** Must be logged in and must have selected this Job by clicking on it on the list of Jobs
- **Event flow description:**

  1. The data is fetched from the Database
  2. The Job details and specifics are displayed to the User
  3. Leaders and Organisers can click the 'Modify this Job details' or the 'Delete this Job' button in order to remove it, taking them to a new Screen
  4. Leaders and Organisers can click the 'see Worker Applications' button taking them to a new Screen
  5. The Administrator can click the 'Moderate this Job' button taking them to a new Screen

  1.a Data not successfully retrieved
    1. An error message is displayed informing the User data couldn't be retrieved
    2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
    3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

UC18 - Modify Job details

- **Main Stakeholders:** Leader, Organiser
- **Goal:** Edit the details and specifics of the selected Job
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in and must have selected this Job by clicking on it on the list of Jobs
- **Event flow description:**

  1. A new Screen is shown to the User featuring all the Job details, but their editing is enabled
  2. The User arbitrarily edits the Job

3. Upon being done, they can press the 'Save changes' button - the changes are sent to the Server

1.a Data not successfully retrieved
   1. An error message is displayed informing the User data couldn't be retrieved
   2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
   3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

3.a Data not successfully sent to the Server
   1. An error message is displayed informing the User data wasn't successfully sent to the Server
   2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
   3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC19 - View the list of all the Festivals

- **Main Stakeholders:** Administrator
- **Goal:** View the list of all the Festivals
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in
- **Event flow description:**
   1. Data is fetched from the Server
   2. A new Screen is shown to the Administrator featuring all the Festivals
   3. The Administrator arbitrarily selects the Festival
   4. They are taken to a new screen where Festival details are displayed - data retrieved from the Server

1.a Data not successfully retrieved
   1. An error message is displayed informing the User data couldn't be retrieved
   2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
   3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

4.a Data not successfully retrieved
1. An error message is displayed informing the User data couldn't be retrieved
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

UC20 - Moderate the Festival

- **Main Stakeholders:** Administrator
- **Goal:** Modify or delete the selected Festival
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in and must have selected the Festival to be moderated
- **Event flow description:**

1. Data is fetched from the Server
2. A new Screen is shown to the Administrator featuring Festival details
3. The Administrator arbitrarily edits the data
4. The Administrator clicks the 'Save changes' button or the 'Delete Festival' button, or he just clicks the backward arrow and is taken to the previous Screen
5. Changes are sent to the Server and saved

1.a Data not successfully retrieved
1. An error message is displayed informing the User data couldn't be retrieved
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

3.a The User enters disallowed/illegal/rubbish data and the Server doesn't accept such an attempt
1. The wrong field values are reset
2. The User is notified that they have entered improper input. The notification mentions the limits and asks the User to change the invalid fields

3. The User enters the data again. If the input is illegal the same steps above apply.
4. Data is accepted and saved this time

UC21 - View the list of own Festivals(Leader, Organiser)

- **Main Stakeholders:** Leader, Organiser
- **Goal:** See the list of all the Festivals that the Leader created, or that the Organiser organises
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in
- **Event flow description:**

1. The User presses the sandwich button and then presses the 'View my Festivals' button
2. Data is fetched from the Server
3. A new Screen is shown, featuring the list of Festivals
4. The Leader/Organiser can tap on the Festival to open up its details

1.a Data not successfully retrieved

1. An error message is displayed informing the User data couldn't be retrieved
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

UC22 - View the Festival details(Leader/Organiser)

- **Main Stakeholders:** Leader, Organiser
- **Goal:** Inspect Festival details and specifics
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected a Festival
- **Event flow description:**

1. Data is fetched from the Server
2. Festival details are displayed to the User
3. The Leader/Organiser can click the 'Modify Festival details' button which will take them to a new Screen

4. The Leader can click the 'Delete the Festival' button. This will take them to a new Screen for Festival deletion.

1.a Data not successfully retrieved
   1. An error message is displayed informing the User data couldn't be retrieved
   2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
   3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

UC23 - Remove the Festival(Leader)

- **Main Stakeholders:** Leader
- **Goal:** Delete their Festival
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected a Festival
- **Event flow description:**
   1. The Leader is asked if they're sure of deleting the Festival
   2. If yes is pressed again, they're asked if they're **REALLY** sure of deleting the Festival
   3. If yes is again pressed, then the Festival is deleted.
   4. All the stakeholders of the Festival(Organiser and Workers) are sent notifications, and their accounts are updated accordingly
   5. Changes are sent to the Server

5.a Data not successfully sent to the Server
   1. An error message is displayed informing the User data wasn't successfully sent to the Server
   2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
   3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC24 - View Festival Job Timeline

- **Main Stakeholders:** Administrator, Leader, Organiser
- **Goal:** Inspect Festival Timeline regarding Jobs. Job overview
- **Stakeholders:** Database, Back-End(Server)

- **Conditions:** Must be logged in, must have selected a Festival and then selected the button for viewing the Festival Job Timeline
- **Event flow description:**
  1. Data is retrieved from the Server
  2. The Festival Job Timeline is displayed to the User
  3. The User can click on a Job to open up its detail and manipulation Screen

  1.a Data not successfully retrieved
    1. An error message is displayed informing the User data couldn't be retrieved
    2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
    3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

UC25 - Remove a Job

- **Main Stakeholders:** Administrator, Leader, Organiser
- **Goal:** Remove the selected Job
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected a Job
- **Event flow description:**
  1. The User is asked if they're sure of deleting the Job
  2. If yes is pressed again, they're asked if they're **REALLY** sure of deleting the Job
  3. If yes is again pressed, then the Job is deleted
  4. All the stakeholders of the Job(Festival Leader, Organiser and affected Workers) are sent notifications, and their accounts are updated accordingly
  5. Changes are sent to the Server

  5.a Data not successfully sent to the Server
    1. An error message is displayed informing the User data wasn't successfully sent to the Server
    2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
    3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC26 - Managing Festival Job Timeline

- **Main Stakeholders:** Administrator, Leader, Organiser
- **Goal:** Organise Job times in an effective manner
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected a Festival and then its Festival Job Timeline
- **Event flow description:**

1. Data is retrieved from the Server
2. The User can add a Job to the Timeline by clicking the 'Add a Job to the Timeline' taking them to a new Screen
3. The User can remove a Job from the Timeline by selecting it on the Timeline and then dragging it out of the Screen - a confirmation dialogue will appear
4. The User can create a new Timeline Branch by double tapping anywhere on the Timeline
5. The User can remove a Branch long pressing anywhere on it, and then on the pop-up menu selecting the 'Remove Branch' option
6. The User can change the Time of a Job or a Branch by either dragging them relative to the Time Axis, or by long pressing and selecting 'Change Time' option
7. The User can change the duration by changing the dimensions of the Job on the graph - elongating or shortening it relative to the time axis
8. Data is sent to the Server, and changes there are saved

1.a Data not successfully retrieved due to connection timeout(can be caused by various network problems)

1. An error message is displayed informing the User data couldn't be retrieved
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

7.a Illegal input(Time format!)

1. An error message is displayed informing the Leader that he has enter an illegal input

2. The notification asks the Leader to re-format and re-enter the input. The expected format and rules are displayed
3. The steps above are repeated if new input isn't accepted either

8.a Data not successfully sent to the Server

1. An error message is displayed informing the User data wasn't successfully sent to the Server
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC27 - Add a Job to the Festival Timeline(Screen)

- **Main Stakeholders:** Leader, Organiser
- **Goal:** Add a Job onto the Festival Timeline Screen - so that it can be furtherly managed, organised and/or deleted
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected a Festival, its Festival Job Timeline and then selected the button 'Add a Job to the Timeline'
- **Event flow description:**

1. The Job is added to the Festival Job Timeline
2. The User is notified of the success
3. Changes are sent to the Server
4. The User is taken back

3.a Data not successfully sent to the Server

1. An error message is displayed informing the User data wasn't successfully sent to the Server
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC28 - View Job Worker Application Entries of the current Festival

- **Main Stakeholders:** Leader, Organiser
- **Goal:** View the list of all the Job Worker Application Entries, so that they can be verified in order for Workers to be selected to work on the given Jobs.

- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected a Festival
- **Event flow description:**

  1. Tap the button to 'View the Worker Job Application Entries'
  2. The list is fetched from the Server
  3. The User is taken to a new Screen where all the entries are displayed
  4. Each Entry has an 'Inspect entry' button - upon pressing it the User it taken to a new Screen

  2.a Data not successfully retrieved due to connection timeout(can be caused by various network problems)

  1. An error message is displayed informing the User data couldn't be retrieved
  2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
  3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

UC29 - Inspect Job Worker Application Entry

- **Main Stakeholders:** Leader, Organiser
- **Goal:** Inspect the Job Entry more closely as to be able to make a more quality decision on whether to verify this Entry or not
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected a Festival and a Job Entry
- **Event flow description:**

  1. Data is fetched from the Server
  2. The User inspects the data
  3. If deemed fit, the User can verify the Entry
  4. The changes are sent to the Server
  5. By tapping the button 'Back' the User can go back to the Job Application Entries List

  1.a Data not successfully retrieved

  1. An error message is displayed informing the User data couldn't be retrieved
  2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again

3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

4.a Data not successfully sent to the Server

1. An error message is displayed informing the User data wasn't successfully sent to the Server
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC30 - Verify Worker Job Application entry

- **Main Stakeholders:** Leader, Organiser, Administrator
- **Goal:** Upon inspection, if the Job Entry is deemed quality, verify this Entry so that it can compete with other Entries for the final approval
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected a Job to verify
- **Event flow description:**

1. Upon clicking the 'Verify this Job Application Entry' this Worker's Entry is verified
2. It enters the Job Auction process
3. Data is sent to the Server

3.a Data not successfully sent to the Server

1. An error message is displayed informing the User data wasn't successfully sent to the Server
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC31 - Manage Job Activities

- **Main Stakeholders:** Worker, Administrator
- **Goal:** Manage and modify the list of all the Activities that make up a Job
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected a Job of which the Activities will be viewed and must have tapped the 'Manage Job Activities' button of the selected Job

- **Event flow description:**
  1. Data(the list) is fetched from the Server and displayed to the User
  2. The User can select a Specific Activity to inspect its Details
  3. The User can add or remove Activities
  4. The User can tap the 'View Job Activities Timeline' in order to manage the time-dimension of the aforementioned Activities

  1.a Data not successfully retrieved
    1. An error message is displayed informing the User data couldn't be retrieved
    2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
    3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

UC32 - Inspect Job Activity details

- **Main Stakeholders:** Worker, Administrator
- **Goal:** Further inspect and view the details of a specific Job Activity
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected a Job and one of its Activities
- **Event flow description:**
  1. Data(Job details) is fetched from the Server
  2. The details are displayed to the User
  3. The User can remove or modify this Activity
  4. If any changes are made, they are sent to the Server

  1.a Data not successfully retrieved
    1. An error message is displayed informing the User data couldn't be retrieved
    2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
    3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

  3.a Data not successfully sent to the Server
    1. An error message is displayed informing the User data wasn't successfully sent to the Server

2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC33 - Modify Job Activity

- **Main Stakeholders:** Worker, Administrator
- **Goal:** Modify Job Activity details
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected a Job and one of its Activities
- **Event flow description:**

1. Data(Job details) is fetched from the Server
2. The details are displayed to the User – editable
3. The User arbitrarily modifies the data
4. If any changes are made, they are sent to the Server

1.a Data not successfully retrieved

1. An error message is displayed informing the User data couldn't be retrieved
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

3.a Illegal input

1. An error message is displayed informing the Leader that he has enter an illegal input
2. The notification asks the Leader to re-format and re-enter the input. The expected format and rules are displayed
3. The steps above are repeated if new input isn't accepted either

4.a Data not successfully sent to the Server

1. An error message is displayed informing the User data wasn't successfully sent to the Server
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC34 - Create Job Activity

- **Main Stakeholders:** Worker, Administrator
- **Goal:** Create a new Job Activity
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have tapped the button to create a new Job Activity for a specific Job
- **Event flow description:**

1. The User fills the form out arbitrarily
2. Upon completion, they tap the 'Create Job Activity'
3. The updated data is sent to the Server and saved

1.a Illegal input

1. An error message is displayed informing the Leader that he has enter an illegal input
2. The notification asks the Leader to re-format and re-enter the input. The expected format and rules are displayed
3. The steps above are repeated if new input isn't accepted either

3.a Data not successfully sent to the Server

1. An error message is displayed informing the User data wasn't successfully sent to the Server
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC35 - Remove Job Activity

- **Main Stakeholders:** Worker, Administrator
- **Goal:** Remove the selected Job Activity
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected a Job and the Activity to be deleted
- **Event flow description:**

1. The User is prompted whether they are sure of the deletion
2. They are again prompted, just in case
3. The Activity is removed, and changes sent to the Server

3.a Data not successfully sent to the Server
   1. An error message is displayed informing the User data wasn't successfully sent to the Server
   2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
   3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC36 - Manage Job Activities Timeline(Worker View)

- **Main Stakeholders:** Worker, Administrator
- **Goal:** Inspect and modify(in Administrator's case moderate) the list of Activities of the given Job
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected the option to go to Job Activity Timeline Management Screen
- **Event flow description:**
   1. Data is fetched from the Server
   2. The time-relative graph is displayed to the User
   3. Job Activity Management offers the same functionalities as the Festival Job Management interface
   4. The User can add a Job to the Timeline by tapping the 'Add a Job to the Timeline' button taking them to a new Screen
   5. The User can remove a Job from the Timeline by selecting it on the Timeline and then dragging it out of the Screen - a confirmation dialogue will appear
   6. The User can create a new Timeline Branch by double tapping anywhere on the Timeline
   7. The User can remove a Branch long pressing anywhere on it, and then on the pop-up menu selecting the 'Remove Branch' option
   8. The User can change the Time of a Job or a Branch by either dragging them relative to the Time Axis, or by long pressing and selecting 'Change Time' option
   9. The User can change the duration by changing the dimensions of the Job on the graph - elongating or shortening it relative to the time axis
   10. When done with changing the diagram, the User can tap the 'Save changes' button - data is sent to the Server and saved there

11.

1.a Data not successfully retrieved
    1. An error message is displayed informing the User data couldn't be retrieved
    2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
    3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

4.a Data not successfully sent to the Server
    1. An error message is displayed informing the User data wasn't successfully sent to the Server
    2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
    3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC37 - Add Job Activity to Timeline

- **Main Stakeholders:** Worker, Administrator
- **Goal:** Add the selected Job Activity to the Timeline so that its Time parameters can be set and manipulated
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, must have selected a Job and the Activity to be added to the Job Activity Timeline
- **Event flow description:**
    1. The User is prompted whether they are sure of the addition
    2. Upon confirmation, the Activity is added to the Timeline and can be manipulated there
    3. The confirmation is sent to the Server in order to be parsed and saved

3.a Data not successfully sent to the Server
    1. An error message is displayed informing the User data wasn't successfully sent to the Server
    2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
    3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC38 - Print Festival Card

- **Main Stakeholders:** Leader, Organiser, and Worker
- **Goal:** Festival Stakeholder can print Cards granting them an entrance to the Festival itself
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in
- **Event flow description:**

1. The User selects one of the Festivals or Jobs that they're responsible for
2. They download the .pdf of the Card from the Server

2.a Data not successfully retrieved

1. An error message is displayed informing the User data couldn't be retrieved
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to fetch data again
3. Upon successful data retrieval(If ever) the button and error message are removed and then the data is displayed

UC39 - Job Entries Auction

- **Main Stakeholders:** Worker
- **Goal:** Select the Workers according to their Job Entries who will be given the task of performing the Job
- **Stakeholders:** Database, Back-End(Server), Leader, and Organiser
- **Conditions:** Must be logged in
- **Event flow description:**

1. After a specified period, according to the Job requirements, one or more Worker's entries are selected
2. The selected Workers are notified
3. The Jobs are added to the Worker's list of Jobs to be performed
4. Data is uploaded to the Server

4.a Data not successfully sent to the Server

1. An error message is displayed informing the User data wasn't successfully sent to the Server
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again

3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC40 - Extend Job Auction by 1 day

- **Main Stakeholders:** Leader, Organiser
- **Goal:** Extend the Job Auction by 1 day due to various reasons - up to Leader/Organiser discretion
- **Stakeholders:** Database, Back-End(Server), Worker
- **Conditions:** Must be logged in
- **Event flow description:**

1. The User taps the button to extend the Auction by 1 day
2. The Auction is extended by 1 day
3. Changes are sent to the Server to be parsed and saved there

3.a Data not successfully sent to the Server

1. An error message is displayed informing the User data wasn't successfully sent to the Server
2. A 'Retry' button is displayed on the screen - upon clicking it the app attempts to send the data again
3. Upon success(if ever) the Screen is closed. Otherwise the aforementioned procedure is again executed

UC41 - View list of Job Auctions

- **Main Stakeholders:** Leader, Organiser, Administrator
- **Goal:** View the list of all the Job Auctions(both on-going and finished)
- **Stakeholders:** Database, Back-End(Server)
- **Conditions:** Must be logged in, and must have selected a Festival of which Job Auctions will be displayed
- **Event flow description:**

1. The User taps the button to view the list of Job Auctions
2. The list of Job Auctions is displayed to the User
3. The User can add, remove or further inspect various Job Auctions

Use Case Diagrams

Prikazati odnos aktora i obrazaca uporabe odgovarajućim UML dijagramom. Nije nužno nacrtati sve na jednom dijagramu. Modelirati po razinama apstrakcije i skupovima srodnih funkcionalnosti.

3.1.2 Sekvencijski dijagrami

dio 1. revizije

Nacrtati sekvencijske dijagrame koji modeliraju najvažnije dijelove sustava (max. 4 dijagrama). Ukoliko postoji nedoumica oko odabira, razjasniti s asistentom. Uz svaki dijagram napisati detaljni opis dijagrama.

3.2 Ostali zahtjevi

dio 1. revizije

Nefunkcionalni zahtjevi i zahtjevi domene primjene dopunjuju funkcionalne zahtjeve. Oni opisuju ***kako se sustav treba ponašati*** *i koja* ***ograničenja*** *treba poštivati (performanse, korisničko iskustvo, pouzdanost, standardi kvalitete, sigurnost...). Primjeri takvih zahtjeva u Vašem projektu mogu biti: podržani jezici korisničkog sučelja, vrijeme odziva, najveći mogući podržani broj korisnika, podržane web/mobilne platforme, razina zaštite (protokoli komunikacije, kriptiranje...)... Svaki takav zahtjev potrebno je navesti u jednoj ili dvije rečenice.*

git

4. Architecture and System Design

dio 1. revizije

Potrebno je opisati stil arhitekture te identificirati: podsustave, preslikavanje na radnu platformu, spremišta podataka, mrežne protokole, globalni upravljački tok i sklopovsko-programske zahtjeve. Po točkama razraditi i popratiti odgovarajućim skicama:

- *izbor arhitekture temeljem principa oblikovanja pokazanih na predavanjima (objasniti zašto ste baš odabrali takvu arhitekturu)*
- *organizaciju sustava s najviše razine apstrakcije (npr. klijent-poslužitelj, baza podataka, datotečni sustav, grafičko sučelje)*
- *organizaciju aplikacije (npr. slojevi frontend i backend, MVC arhitektura)*

4.1 Database

Database used for this project is a relation based database. Relation is usually referred to as a table that has tuples. Tuple is an object that represents an information. Purpose of the database is easy and fast data manipulation, including saving, deleting, updating and sending data to the server. Database has relations:

- *User*
- *Festival*
- *Event*
- *Specialization*
- *WorkerSpec*
- *Auction*
- *Application*
- *Job*

- *JobSpec*
- *FestivalOrganizers*

4.1.1 Tables details

User *has entitites for every usse er of the app. It has all needed personal information about the user and his role.*

User		
user_id	INT	User identification number
username	VARCHAR	Unique username
password	VARCHAR	User account password
firstname	VARCHAR	Users first name
lastname	VARCHAR	Users last name
picture	VARCHAR	Profile picture string
phone	VARCHAR	Users phone number
email	VARCHAR	Users email address
role	VARCHAR	Role user will persue in application

Festival *contains all needed information about the festival.*

Festival		
festival_id	INT	Festival identification number
creator_id	INT	ID of the user who created the festival
name	VARCHAR	Festival name
desc	VARCHAR	Short festival description, can be empty
logo	VARCHAR	Festivals logo string
duration	INTERVAL	Festival duration interval
active	BOOLEAN	True if festival is active, False if it's unactive

Event *contains information about the event of the festival.*

Event		
event_id	INT	Event identification number
festival_id	INT	ID of the festival that event belongs to
organizer_id	INT	ID of the user who created the event

Event		
name	VARCHAR	Event name
desc	VARCHAR	Short event description, can be empty
location	VARCHAR	Events location
start_Time	TIMESTAMP	Event start time
end_Time	TIMESTAMP	Event end time

Specialization *contains all different specializations and their names.*

Specialization		
specialization_id	INT	Specializations identification number
name	VARCHAR	Name of the specialization

WorkerSpec *contains specifications about the user who wants to apply as a worker and his specializations.*

WorkerSpec		
worker_id	INT	Workers identification number
specialization_id	INT	Specializations identification number

Auction *contains information about auctions.*

Auction		
auction_id	INT	Auction identification number
start_Time	TIMESTAMP	Auction start time
end_Time	TIMESTAMP	Auction end time

Application *contains information about applications for auctions.*

Application		
application_id	INT	Application identification number
auction_id	INT	Auction identification number that worker applies for
worker_id	INT	Workers identification number
price	FLOAT	Offered pay for the job
comment	VARCHAR	Additional comment for application, can be empty

Application		
approximate_time	INT	Time needed to complete the job, in days
number_of_people	INT	Number of people that will be doing the job

Job *contains information about jobs that had an auction.*

Job		
job_id	INT	Job identification number
event_id	INT	Events identification number that job is for
worker_id	INT	Workers identification number that does the job
auction_id	INT	Auctions identification number that auctioned the job
start_Time	DATETIME	Jobs start time
is_Completed	BOOLEAN	True if job is finished, false if it's still active

JobSpec *contains specializations that are needed for the job.*

JobSpec		
job_id	INT	Job identification number
specialization_id	INT	Specializations identification number

FestivalOrganizers *contains information which user, that is also an organizer, applied for which festival.*

FestivalOrganizers		
festival_id	INT	Festival identification number
organizer_id	INT	Organizers identification number
status	INT	1 when organizer is waiting on leader, 0 when rejected and 1 when accepted

4.1.2 Database diagrams

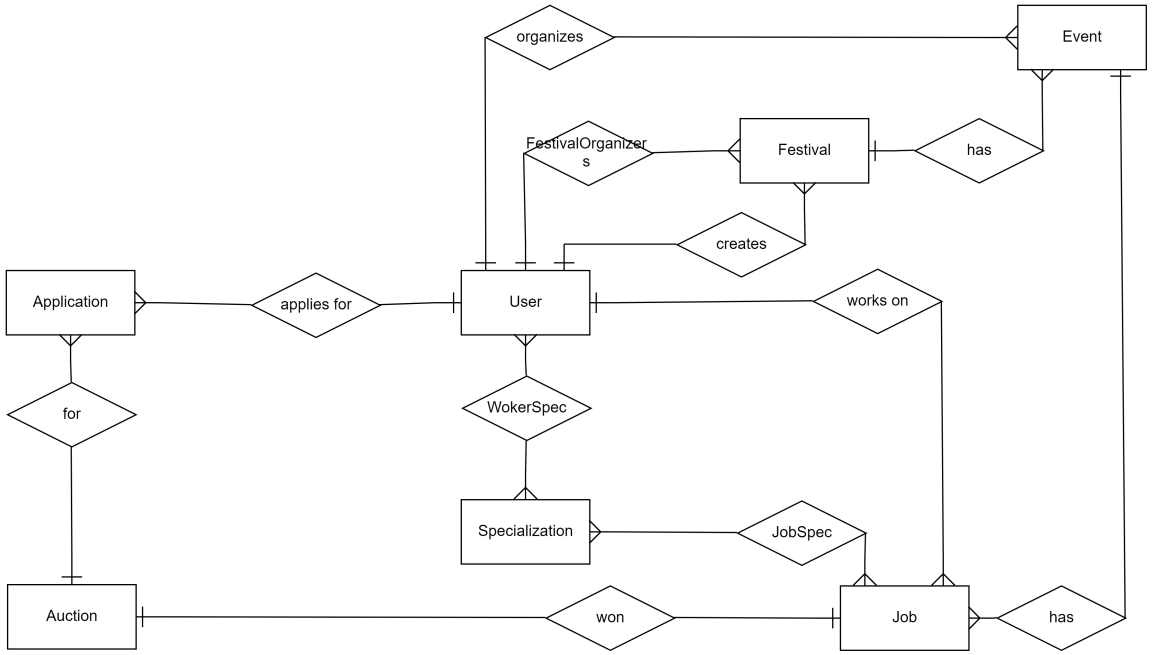

Figure 4.1: E-R Diagram

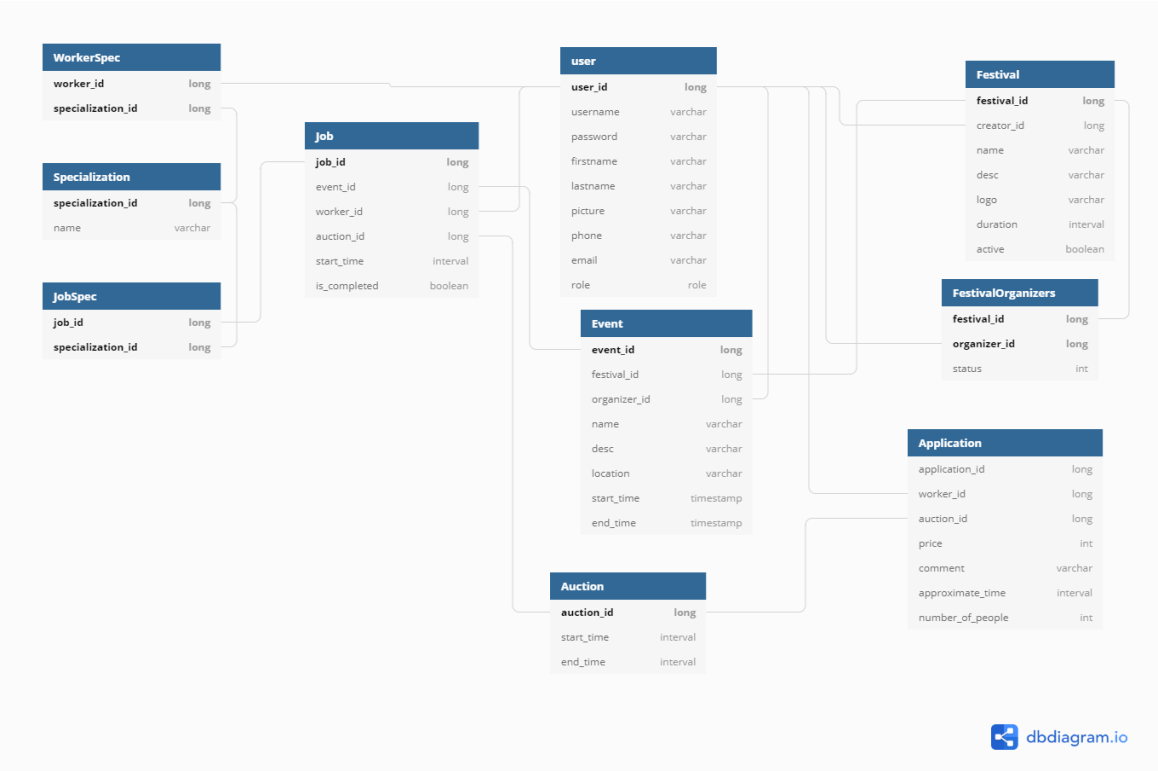

Figure 4.2: Database Diagram

4.2 Class Diagram

Potrebno je priložiti dijagram razreda s pripadajućim opisom. Zbog preglednosti je moguće dijagram razlomiti na više njih, ali moraju biti grupirani prema sličnim razinama apstrakcije i srodnim funkcionalnostima.

dio 1. revizije

*Prilikom prve predaje projekta, potrebno je priložiti potpuno razrađen dijagram razreda vezan uz **generičku funkcionalnost** sustava. Ostale funkcionalnosti trebaju biti idejno razrađene u dijagramu sa sljedećim komponentama: nazivi razreda, nazivi metoda i vrste pristupa metodama (npr. javni, zaštićeni), nazivi atributa razreda, veze i odnosi između razreda.*

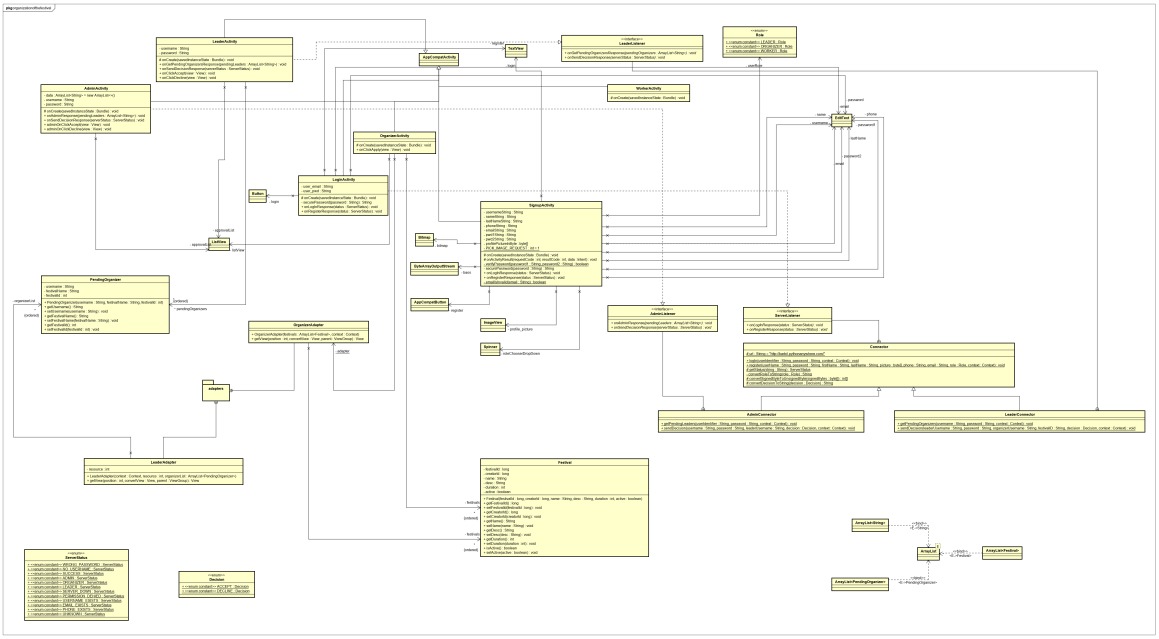

Figure 4.3: Class Diagram

dio 2. revizije

Prilikom druge predaje projekta dijagram razreda i opisi moraju odgovarati stvarnom stanju implementacije

4.3 Dijagram stanja

dio 2. revizije

*Potrebno je priložiti dijagram stanja i opisati ga. Dovoljan je jedan dijagram stanja koji prikazuje **značajan dio funkcionalnosti** sustava. Na primjer, stanja korisničkog sučelja i tijek korištenja neke ključne funkcionalnosti jesu značajan dio sustava, a registracija i prijava nisu.*

4.4 Dijagram aktivnosti

dio 2. revizije

Potrebno je priložiti dijagram aktivnosti s pripadajućim opisom. Dijagram aktivnosti treba prikazivati značajan dio sustava.

4.5 Dijagram komponenti

dio 2. revizije

Potrebno je priložiti dijagram komponenti s pripadajućim opisom. Dijagram komponenti treba prikazivati strukturu cijele aplikacije.

5. Implementation and User Interface

5.1 Korištene tehnologije i alati

dio 2. revizije

Detaljno navesti sve tehnologije i alate koji su primijenjeni pri izradi dokumentacije i aplikacije. Ukratko ih opisati, te navesti njihovo značenje i mjesto primjene. Za svaki navedeni alat i tehnologiju je potrebno ***navesti internet poveznicu*** *gdje se mogu preuzeti ili više saznati o njima.*

5.2 Ispitivanje programskog rješenja

dio 2. revizije

U ovom poglavlju je potrebno opisati provedbu ispitivanja implementiranih funkcionalnosti na razini komponenti i na razini cijelog sustava s prikazom odabranih ispitnih slučajeva. Studenti trebaju ispitati temeljnu funkcionalnost i rubne uvjete.

5.2.1 Ispitivanje komponenti

*Potrebno je provesti ispitivanje jedinica (engl. unit testing) nad razredima koji implementiraju temeljne funkcionalnosti. Razraditi **minimalno 6 ispitnih slučajeva** u kojima će se ispitati redovni slučajevi, rubni uvjeti te izazivanje pogreške (engl. exception throwing). Poželjno je stvoriti i ispitni slučaj koji koristi funkcionalnosti koje nisu implementirane. Potrebno je priložiti izvorni kôd svih ispitnih slučajeva te prikaz rezultata izvođenja ispita u razvojnom okruženju (prolaz/pad ispita).*

5.2.2 Ispitivanje sustava

Potrebno je provesti i opisati ispitivanje sustava koristeći radni okvir Selenium[1]*. Razraditi **minimalno 4 ispitna slučaja** u kojima će se ispitati redovni slučajevi, rubni uvjeti te poziv funkcionalnosti koja nije implementirana/izaziva pogrešku kako bi se vidjelo na koji način sustav reagira kada nešto nije u potpunosti ostvareno. Ispitni slučaj se treba sastojati od ulaza (npr. korisničko ime i lozinka), očekivanog izlaza ili rezultata, koraka ispitivanja i dobivenog izlaza ili rezultata.*

Izradu ispitnih slučajeva pomoću radnog okvira Selenium moguće je provesti pomoću jednog od sljedeća dva alata:

- *dodatak za preglednik **Selenium IDE** - snimanje korisnikovih akcija radi automatskog ponavljanja ispita*
- ***Selenium WebDriver** - podrška za pisanje ispita u jezicima Java, C#, PHP koristeći posebno programsko sučelje.*

Detalji o korištenju alata Selenium bit će prikazani na posebnom predavanju tijekom semestra.

[1] `https://www.seleniumhq.org/`

5.3 Dijagram razmještaja

dio 2. revizije

*Potrebno je umetnuti **specifikacijski** dijagram razmještaja i opisati ga. Moguće je umjesto specifikacijskog dijagrama razmještaja umetnuti dijagram razmještaja instanci, pod uvjetom da taj dijagram bolje opisuje neki važniji dio sustava.*

5.4 Upute za puštanje u pogon

dio 2. revizije

U ovom poglavlju potrebno je dati upute za puštanje u pogon (engl. deployment) ostvarene aplikacije. Na primjer, za web aplikacije, opisati postupak kojim se od izvornog kôda dolazi do potpuno postavljene baze podataka i poslužitelja koji odgovara na upite korisnika. Za mobilnu aplikaciju, postupak kojim se aplikacija izgradi, te postavi na neku od trgovina. Za stolnu (engl. desktop) aplikaciju, postupak kojim se aplikacija instalira na računalo. Ukoliko mobilne i stolne aplikacije komuniciraju s poslužiteljem i/ili bazom podataka, opisati i postupak njihovog postavljanja. Pri izradi uputa preporučuje se ***naglasiti korake instalacije uporabom natuknica*** *te koristiti što je više moguće* ***slike ekrana*** *(engl. screenshots) kako bi upute bile jasne i jednostavne za slijediti.*

Dovršenu aplikaciju potrebno je pokrenuti na javno dostupnom poslužitelju. Studentima se preporuča korištenje neke od sljedećih besplatnih usluga: Amazon AWS, Microsoft Azure ili Heroku. Mobilne aplikacije trebaju biti objavljene na F-Droid, Google Play ili Amazon App trgovini.

6. Conclusion and Outline of Planned Future Work

dio 2. revizije

U ovom poglavlju potrebno je napisati osvrt na vrijeme izrade projektnog zadatka, koji su tehnički izazovi prepoznati, jesu li riješeni ili kako bi mogli biti riješeni, koja su znanja stečena pri izradi projekta, koja bi znanja bila posebno potrebna za brže i kvalitetnije ostvarenje projekta i koje bi bile perspektive za nastavak rada u projektnoj grupi.

Potrebno je točno popisati funkcionalnosti koje nisu implementirane u ostvarenoj aplikaciji.

Literature

Kontinuirano osvježavanje

Popisati sve reference i literaturu koja je pomogla pri ostvarivanju projekta.

1. Oblikovanje programske potpore, FER ZEMRIS, http://www.fer.hr/predmet/opp
2. I. Sommerville, "Software engineering", 8th ed, Addison Wesley, 2007.
3. T.C.Lethbridge, R.Langaniere, "Object-Oriented Software Engineering", 2nd ed. McGraw-Hill, 2005.
4. I. Marsic, Software engineering book", Department of Electrical and Computer Engineering, Rutgers University, http://www.ece.rutgers.edu/~marsic/books/SE
5. The Unified Modeling Language, https://www.uml-diagrams.org/
6. Astah Community, http://astah.net/editions/uml-new

Image and diagram index

4.1 E-R Diagram 50
4.2 Database Diagram 51
4.3 Class Diagram 52

Dodatak: Prikaz aktivnosti grupe

Meeting log

Kontinuirano osvježavanje

1. Meeting
   - Date: October 4, 2019
   - Attented: P.Fribert, D.R.Sparemblek, B.Bilic, K.Ceple, D.Strbad, L.Lendel, K.Jezic
   - Meeting theme:
     - initial getting to know each other, individual skills
2. Meeting
   - Date: October 5, 2019
   - Attented: P.Fribert, D.R.Sparemblek, B.Bilic, K.Ceple, D.Strbad, L.Lendel, K.Jezic
   - Meeting theme:
     - creating back end, front end and server team
3. Meeting
   - Date: October 15, 2019
   - Attented:D.R.Sparemblek, B.Bilic
   - Meeting theme:
     - making first draft database
4. Meeting
   - Date: October 23, 2019
   - Attented:D.R.Sparemblek, B.Bilic, P.Fribert
   - Meeting theme:
     - making database in python
5. Meeting
   - Date: October 24, 2019
   - Attented:L.Lendel, B.Bilic, K.Jezic, D.R.Šparemblek, P.Fribert

- Meeting theme:
  - signup and login screen functions

6. Meeting
   - Date: November 6, 2019
   - Attented: P.Fribert, D.R.Sparemblek
   - Meeting theme:
     - foreign keys in database
7. Meeting
   - Date: November 7, 2019
   - Attented: P.Fribert, B.Bilic, K.Ceple, D.Strbad, L.Lendel, K.Jezic
   - Meeting theme:
     - server functions
     - front end: hashing passwords and checking emails
     - documentation update
8. Meeting
   - Date: November 10, 2019
   - Attented: D.R.Sparemblek, B.Bilic, L.Lendel, K.Jezic
   - Meeting theme:
     - debugging application, checking code
9. Meeting
   - Date: November 11, 2019
   - Attented: P.Fribert, D.R.Sparemblek, B.Bilic, K.Ceple, D.Strbad, L.Lendel, K.Jezic
   - Meeting theme:
     - final changes before commiting project
10. Meeting
    - Date: November 13, 2019
    - Attented: P.Fribert, K.Ceple
    - Meeting theme:
      - documentation specifics

Tablica aktivnosti

Kontinuirano osvježavanje

Napomena: Doprinose u aktivnostima treba navesti u satima po članovima grupe po aktivnosti.

	Ime Prezime voditelja	**Ime Prezime**	**Ime Prezime**	**Ime Prezime**	**Ime Prezime**	**Ime Prezime**	**Ime Prezime**
Upravljanje projektom	x	x					
Opis projektnog zadatka	x	x					
Funkcionalni zahtjevi	x						
Opis pojedinih obrazaca							
Dijagram obrazaca							
Sekvencijski dijagrami							
Opis ostalih zahtjeva							
Arhitektura i dizajn sustava	x	x					x
Baza podataka	x	x					x
Dijagram razreda							
Dijagram stanja							
Dijagram aktivnosti							
Dijagram komponenti							
Korištene tehnologije i alati	x						
Ispitivanje programskog rješenja	x	x			x		x
Dijagram razmještaja							
Upute za puštanje u pogon							
Dnevnik sastajanja							x
Zaključak i budući rad							
Popis literature							

	Bartol Bilic	**Daniel Rey Sparemblek**	**Karlo Jezic**	**Danijel Strbad**	**Luka Lendel**	**Kristijan Ceple**	**Petra Fribert**
front end	x	x			x		
izrada baze podataka	x	x					x
spajanje s bazom podataka	x	x	x		x		
back end							

Dijagrami pregleda promjena

dio 2. revizije

Prenijeti dijagram pregleda promjena nad datotekama projekta. Potrebno je na kraju projekta generirane grafove s gitlaba prenijeti u ovo poglavlje dokumentacije. Dijagrami za vlastiti projekt se mogu preuzeti s gitlab.com stranice, u izborniku Repository, pritiskom na stavku Contributors.