

Changed parts marked with black bold font, you can go through the links in the content !!!

Din-Din - Social dinner organizer

Design Document

Course: CS-E4400 - Design of WWW-services

Project primary focus area: UI and usability

Project secondary focus area: dynamic functionality

Group:

Aleksi Jokela, 293215, aleksi.o.jokela@aalto.fi

Anastasiia Karpenko, 604820, anastasiia.karpenko@aalto.fi

Irina Ziborova, 584869, irina.ziborova@aalto.fi

Timo Övermark, 605382, timo.overmark@aalto.fi

Table of contents

Design Document	0
Table of contents	1
Purpose of the web-service	3
Background research	4
Survey	4
Interviews	4
Competitor analysis and benchmarking	4
The Users	7
Personas	7
Scenarios / Use Cases	9
Scenario 1 - Breaking boundaries	9
Scenario 2 - Reconnecting with old contacts	9
Scenario 3 - The host	9
User paths	10
Stakeholders	11
Design	12
Requirements	12
Site Structure	13
Maintenance and administration	14
Managing content copyrights	15
User interface	16
Sketches and descriptions of the UI	16
Navigation and Interaction	16
Fonts and colors	17
Technologies	19
Technical outline	19
Back end	19
Front end	19
Dynamic functionalities	20
Back end & Database	21
Database structure	21
Security	23
Project Design	24
Tasks	24
Project structure	25
Initial survey	25
Interviews	25
Competitor analysis / benchmarking	26

Requirements	26
User Interface Design	26
Interface design changes	26
Functional prototype implementation	27
Functional prototype user testing	27
Scenario of user test	28
Final implementation	28
Appendix	30
Appendix 1: User survey	30
Social dinner organizer	30
Appendix 2. Early prototype screenshots	32
Appendix 3. Plan for design changes	35
Appendix 4 Final design	36

Purpose of the web-service

Being a student is a wonderful part in everyone's life. It's full of new experiences, meetings and learning new things. Living on campus and being surrounded by fellow students can be a completely new experience for a freshman who just moved to the campus or a foreigner who came to study for his/her degree program. Everything good is best when shared. Because of this we want to help students find new friends and share these experiences.

On campus there are a lot of events that aim to organize social activities and entertainment for students. However, from our own experience getting to meet new people on campus is still challenging. Most friend-making activities are limited to the timeline of a particular event, such as a drinking and dancing party. When the event is over the students should organize other events with their new friends by themselves. It is so easy to lose newly acquired connections, get back to your old routines still remaining lonely.

In order to encourage the process of making new friends and getting to know new people we would like to design a web-service that would connect people over social meals. Sharing a meal together is a very natural part of human social life. Food is a part of our culture, traditions and a way to get to know people around you better, to familiarize yourself with others' life and reality.

On campus the student canteens are opened only for lunch and the lunch time is quite busy and has a limited period of time. Students have to stay in the queue for some time and then have to eat quickly to get back to their study routines and schedules. There is not much time for socializing and meeting new friends. However, during the dinner time the majority of students who live on campus cook and eat alone in their own rooms/ student flats or dormitory kitchens. We want to use this valuable opportunity to offer a service for meeting new friends and tasting new food within the student society on campus. This will solve the problem that some of the students face:

- They want to meet new friends but don't know when and where?
- They don't want to eat alone and want to share dinner with someone once in awhile.
- In the dormitory kitchen they always eat alone and are not familiar with their neighbors.
- They want to taste new food but find restaurants expensive.
- They want to know more about new cultures and explore foreign traditions.
- They would like to meet international friends and practice new languages.

We want to gather students together in dormitory kitchens over nice social dinners. Our social dinner organizer service is a student focused service targeting over 2000 students in Teekkari kylä Student Village¹. We want to help people meet new friends by sharing food, cooking together and enjoying a dinner together. Our platform helps people to organize group dinners, to search and register these kind of events. The events themselves give the

¹ Wikipedia, "Aalto University" https://en.wikipedia.org/wiki/Aalto_University#Main_Campus_Otaniemi [Accessed on 21.09.2016]

users an opportunity to find new friends, taste international or even exotic food and explore new cultures or practice language skills with exchange students. It will offer an alternative social opportunity to the parties, clubbing and related events for those students that would like to be in a comfortable calm cozy environment.

Background research

Survey

Initial user data was collected using an electronic questionnaire. The questionnaire was sent to target group students. The questions include basic background data from the user group as well as opinions related to dinner organizing. The gathered data has been used to construct personas. Complete survey and statistics of results can be found in the appendix.

The survey was sent to fellow students through common online communication channels, e.g. Facebook and Telegram groups. 17 answers were obtained. The majority of students represent the group of international students who arrived to Otaniemi less than 6 months ago. A few Finnish students who are more or less locals also answered the survey. The majority of responders are interested in using the web-service. Some of them would like to use the service only as guests at dinners organized by the others but more than 60% of responders would like to host the others and cook for the others. The majority of the potential guests would also like to participate in cooking with the host. The main information that the potential guest would like to know about the event is: location and time, info about the host, info about the food and who are the other guests (male/female ratio was also specifically mentioned). For the safety reasons guests would like to have a positive impression on the host (also knowing his hobbies as an option) and make sure nothing bad will happen to them and that they would enjoy their time at the dinner. The potential hosts would like to know some details about the guests. Major details are: personality of the guest and his student status, dietary restrictions and willingness to help with cooking.

Interviews

User data collection will be done by interviewing potential users of the service after the prototype is ready. We will test the prototype and decide on the additional requirements and functions. At the same time we can validate our personas and current understanding of our target group.

Competitor analysis and benchmarking

In this chapter a few similar websites are reviewed to conduct a formal analysis regarding what features they have, what is especially good in the service and what limitations they have. There are a couple of related services already existing online. Two of them in particular are similar to our concept and they are analysed as competitors: VizEat and EatWith. The main difference between Din-Din service and these two services is that Din-Din is designed especially for students. The other two services are mainly targeted touristic “foody” experience and are linked with the locations (touristic centres - attractive

cities around the world). In addition they strive for a monetary transaction based model, whereas Din-Din is a social platform only.

1. Competitor: VizEat, <https://www.vizeat.com/>

Main page

The highest position on the main page is dedicated to the short menu that contains: logo of the company, sign up/log in buttons and the bigger button "post an event".

The main page shows the big introductory photo on the background of search field. The field takes location as a parameter, so the user can browse related events according to the city. Then the next information block shows the most popular destinations listed on the web-site. The following block shortly explains how the service works, advertises their mobile application with the call to download it and then there is the footer with typical contact information and 2nd menu.

The search

When you type the location where you want to search the event, the new pages opens with more advanced search panel and the list of events with pictures. Search panel contains such fields as date (from - to), quantity of guests specifications, selection of the price range for the dinner. You can also load more filters (or hide them, there is a special button for that). You can also specify the language of the host, some details about preferred accommodation, type of event (aperitif, brunch, dinner etc), cuisine type, special diet. You can also filter the events using the geolocation (putting the location on the map of the city that you searched).

Login/Sign up

In order to sign up or login, you can use Facebook or Google+ account or just use your email and password. The process of registration is very fast, easy and clear. Later, after the account is created, you can update it listing more information about you, about your travel and food preferences.

User logged in

When you are logged in, the service shows you another menu on the top: you can access your dashboard, your user profile, private messages and other details that are organized in different 'folders'.

-> Dashboard

Dashboard is not really working and there is not so much sense in it while you are not active user and do not have any event soon.

-> Profile

When you enter your profile, there is a left side menu available, from where you can access and edit personal information of your profile, manage the gallery of your pictures (personal photos and photos of food). What is interesting here is that you location is automatically defined from Google maps (your location from your Facebook/Google profile). All the info that you have on you Facebook/ Google profile will be also saved automatically in your user profile on VizEat. You can also specify

your preferences about the cuisine that you want to cook (not only to eat). There is a predetermined list of food allergies and other food specifications (halal, vegetarian etc) that you can add to your profile. In the profile area you can also see the reviews that you wrote or that someone else has written about you. All the reviews will also be visible on your public profile.

-> Inbox

In this area you can see the conversations that you have: with your hosts and with your guests (shown all together or separately according to your preferences).

-> My Events

Here all the event are visible. If there is no event, there is a button "create an event".

-> I'm a host - I'm a guest

There are the lists of your events. Look at which events you are going to be the host (also past/cancelled/pending requests), or where you are going to be the guest (also past/cancelled/pending requests).

-> Account

There is also a place where you can edit your security data (password and login email) and payment details.

The event page

The event page has its own menu with anchor buttons. The buttons show info about: details of event, about the host, menu, previous reviews, the address and geoposition on the map. On the right side there is a fixed window that will allow to book the event - send a booking request. There is the price of event per person, the field where you can specify the date of the event (the guest picks the date, and the host approves), and the number of guests (up to 3).

Create-an-event - page

In order to create an event the user should push the button in the top right corner "Post an event". On the create-an-event page the user should specify all the information reflected in the advanced search fields. The user can specify minimum number of participants and limit the maximum number of participants.

Design of the site

The design is very clear and clean, the web-site have a good balance of color- empty space that create the sense of lightness and free space. The main color of the site is warm red, that can be associated with food and cozyness.

Important features

There is a good balance between the fields in the search. The user can specifically search for events with such details as dietary restrictions or number of participants.

Constraints

It seems that this web-site is not a community web-site. It seems that users can book seats in an event for himself and his relatives/friends and that it is not possible to book only one place at the event and wait for other people to book the rest of the places. Therefore the experience of this service is limited to food and tourism experience where you get to know only the host as a new connection. However, you cannot make friends with other new people because the guests are you and your friends/relatives. The events do not provide the deadline for registration because basically the users should discuss the date with the host.

2. Competitor: EatWith, <https://www.eatwith.com/>

EatWith is a similar web-service as the previous one. It is also targeted to give tourists a dinner experience. You can search for events by location, type of food, date, price etc. User registration is possible with email and Facebook profile. The main difference from the previous service is that you can also see who is attending the event and check the profile of the guests.

The service also makes suggestions for you to try other events in the area according to your preferences. To book an event, you have to insert the details of your bank card or Paypal account.

Ideas for Din-Din service

The features that seem easiest to adopt to Din-Din are:

- Advanced search fields (dietary restrictions and number of guests),
- Opportunity to specify the limit for guests,
- Opportunity to view the already registered guests for the event and connect to them (as an option).
- It will be better to make the host decide about the date and time of the event and set the deadline for the registration for an event.

The Users

As discussed earlier our service is focused around one specific user group: students. This is because we understand them well, and can focus on learning and creating an good user experience for them. That said - we still need to explore and learn a lot about our users. Even though students often are more or less the same age group and have similar interests, a service targeting a rather sensitive topic, loneliness, needs to have a good connection with its users.

Personas

The service already has a quite tightly defined target user: students. As such we think our target demographic can be condensed into just two personas: Alex and Maija. They are described in the persona posters below.

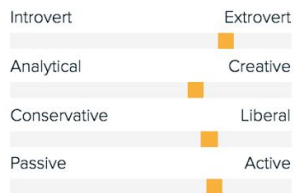
Alex



"I am open to new people."

Age: 19-26
Work: International master student
Family: Single
Location: Espoo
Character: Outgoing

Personality



Friendly Curious Wants to share
Hospitable

Goals

- To meet new friends
- To share his cooking skills
- To have nice time in the company of new friends
- To visit a new friend and taste nice new food

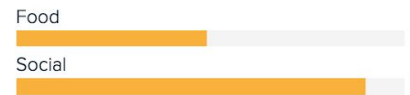
Frustrations

- Spending time with boring people at the dinner

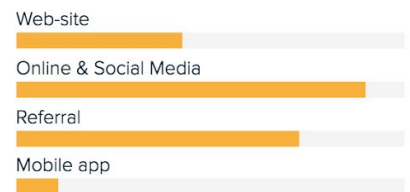
Bio

Alessandro as a foreign student coming to Aalto University to study. He does not know too much about Finland and is still new on campus and wants to get to know more new people. He is curious and opened to new experiences and new friendships. He wants to host and be hosted in a dinner event with nice, interesting people and have a great enjoyable time eating together (maybe cooking together).

Motivations



Preferred Channels



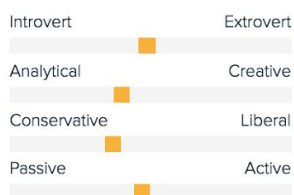
Maija



"I want to know who is attending, where, when, how to get there, who is the person, is he/she a registered sex offender"

Age: 19-26
Work: master student
Family: Single
Location: Espoo
Character: Friendly

Personality



Friendly Foody Curious
Safety conscious

Goals

- To meet new people
- To taste good food
- To have nice evening with new people

Frustrations

- If she feels unsafe in a company of new people
- If she dislikes the food
- If she feels bored with new people

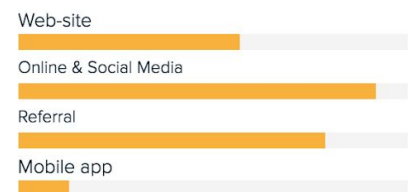
Bio

Maija is a Finnish master student in Otaniemi. She has been living in Otaniemi already for 3 years but still is very interested in meeting new friends and tasting new food. She would prefer to be a guest at a dinner with a company of nice people. She would like to know beforehand about the host, maybe look at his Facebook profile and make sure he/she is a nice decent person, maybe they even have common Facebook friends.

Motivations



Preferred Channels



Scenarios / Use Cases

Scenario 1 - Breaking boundaries

Alex has lived in Otaniemi for a few months now. However, he has come to the inevitable conclusion that Finnish people are shy. He's having a hard time chatting people up and hanging out like he is used to with his friends back home. One Tuesday evening he is ignoring homework and reading social media when he notices a link to Din-Din. He opens it up on his MacBook and is intrigued - finally a channel to get introduced to people outside his current circles.

Alex creates an account and shifts through some of the events, looking for likeminded people. He's interested but not sure if just joining a random dinner would be awkward. Then he notices a dinner request that is in the same house he's living in. "What can I lose" he thinks, if the dinner is awkward he can just walk back home. Alex applies as a guest. Later in the evening he gets accepted. Ready to have an adventure Alex takes a bottle of Tempranillo he got from his mother and heads to his neighbor's place.

Scenario 2 - Reconnecting with old contacts

It's a Friday evening and Maija is coming home from school. She doesn't have any plans for the weekend but she read on Jodel that there's a new event platform called Din-Din. Maija likes the fact that she can meet new people without getting drunk. Unlike most Finns she doesn't enjoy drinking. Hangovers make studying hard.

When she gets home she looks at the events on Din-Din. She notices that an acquaintance of her is hosting a girls night. This seems pretty entry level, she thinks. The dinner theme is left-over tortillas. Everyone can bring whatever they have in their fridge. Low effort and convenient!

Maija has a great time during the dinner. After the dinner she comes back home and endorses the host on Din-Din for going through the effort of organizing the dinner.

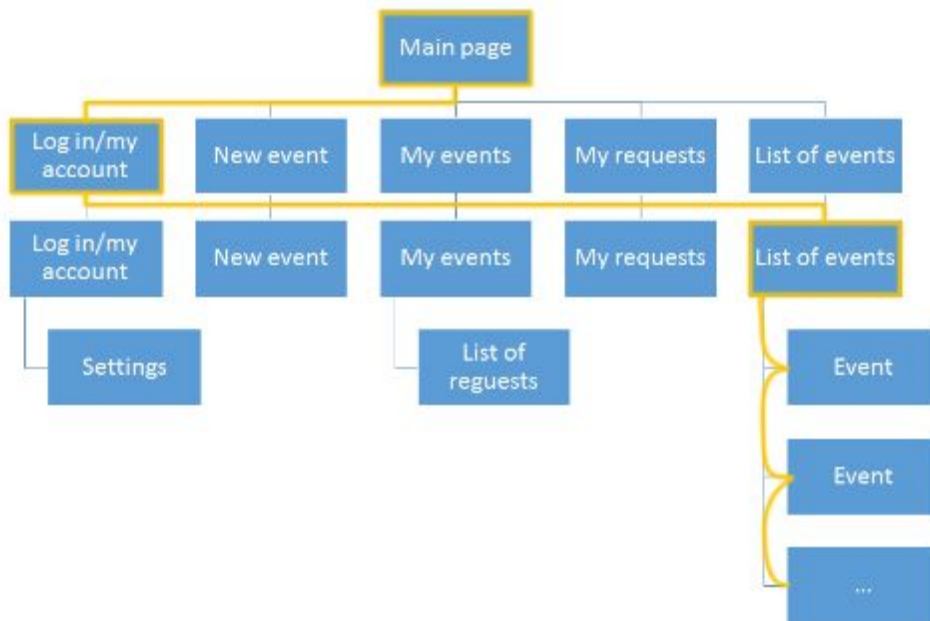
Scenario 3 - The host

After his positive first experience Alex is ready to host his own dinner. He goes to Din-Din and sets up an event. Based on last time he's had enough awkward interactions with the local people. This time he'd rather set up a dinner for international students.

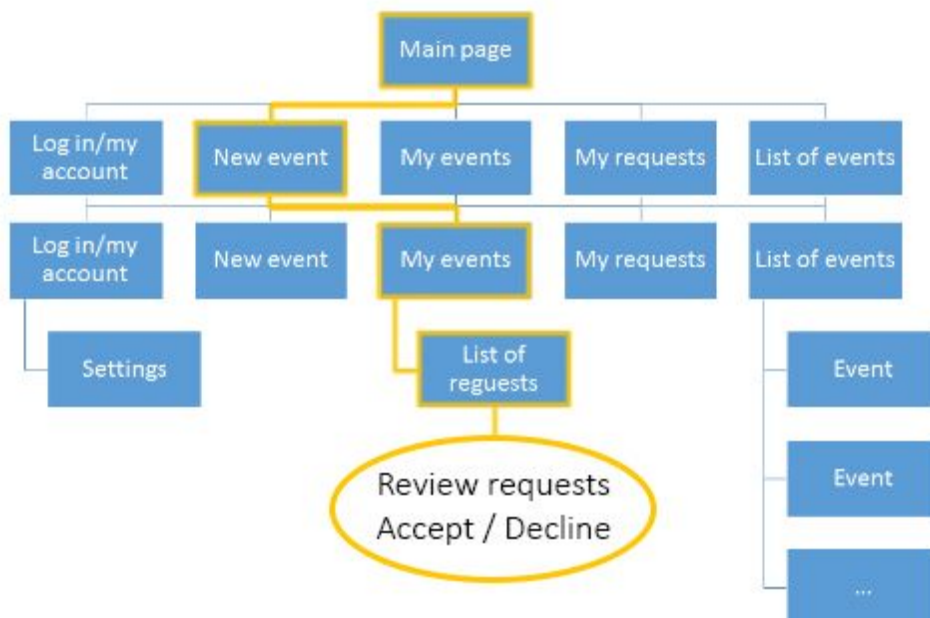
He's a bit concerned if anyone comes so he asks a few of his friends to apply to his dinner as well. His friends can easily join through the service as well. Alex convinces them to start building up their reputation as polite and enjoyable dinner guests by registering immediately.

User paths

Our scenarios are based on two basic user paths - looking for events and creating events. We expect these to be the two most common actions and the user paths for them are shown below. Other noteworthy user paths that we should explore are rating users and finding information about other users. However, these user paths will be more defined after user interviews are conducted and a stronger understanding of how people use the service is built.



User path 1 - browsing through available events.



User path 2 - creating event and reviewing invitations.

Stakeholders

At the beginning of the project we will concentrate only on 2 types of stakeholders who both consist of students. Further development of the business idea of the project can lead to additional stakeholders as advertisers (guilds, student societies, food shops and other event organizers who would want to advertise their events on the Social student dinner organizer page).

Stakeholder	Description
<i>Dinner organizers</i>	Dinner organizers are the students who like or would like to cook not only for themselves. They are very opened and curious about other people also they have something to share about their own culture and they food. They are not necessarily good chefs or may not have professional experience but the big desire to share their food with others. They would like to make new friends and share their own cooked food with them. They prefer to eat in company and don't mind to invite fellow students for a dinner. They also would like to get some kind of reward in addition to opportunity of making new friends: it can be a very small fee covering the expenses for food (1-2 euro) or some extra food or drinks that can supplement the original menu of the dinner. They would want to know a bit of information about possible guests of their event, decide about the limit of participants and their gender (only girls, only boys company, mixed gender). They would want to have an opportunity to approve the members or open the event to anyone who would decide to join.
Goals	
<ul style="list-style-type: none"> • To share dinner with other students • To share cooking experience with other students • To share food expenses for dinner • To organize a dinner event • To approve participants for the dinner event among the applicants 	
<i>Dinner guests</i>	Dinner guests are the students who would like to participate in a dinner with other students. They would want to taste new

	<p>food or just have a company sharing a meal with some new people in a cozy atmosphere. They would want to reward the host by bringing some extra food/delicacies/drinks as a contribution to the dinner or pay 1-2 euro to cover expenses for food or help with cooking process and at the same time learn how to cook new meals. These people are interested in knowing new people, new cultures and traditions. They would want to book the participation in cooking/dinner event online in advance and get the notifications and location (instructions how to get there). They would want to know in advance the menu of the dinner and some details about the dinner organizer.</p>
Goals	
<ul style="list-style-type: none"> • To choose which type of food to eat • To choose the location of dinner event according to the preferences • To participate in a group dinner with other students • To participate in cooking activity • To choose how to reward the dinner organizer • Have an opportunity to choose how to pay/reward • To pay/reward for a low (reasonable) cost • To get information about event in advance • To get information about an event organizer in advance • To get exact location after registration to an event 	

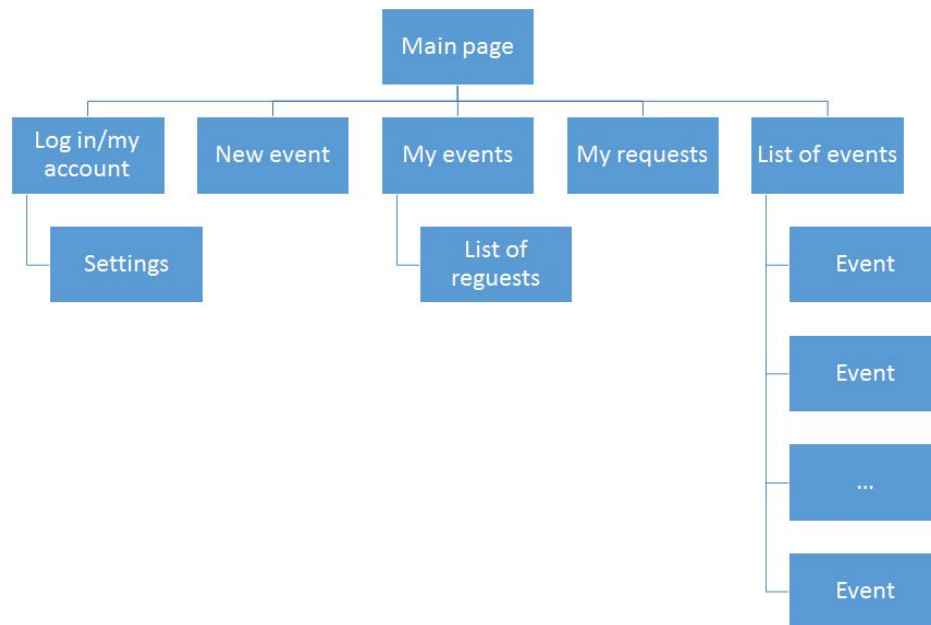
Design

Requirements

- It must be possible to sign up using either a Facebook account or an email address
- User must be able to describe him/herself using a profile picture, user name and a text description
- User must be able to create a dinner event, which is described using a picture, text, address, location on map and categories/tags
- Users can request participation to an event, and event's host may approve or decline the request. Before approving, the exact location (apartment number) of the event is hidden

- Users may post comments on the event. The user can remove his/her own comments, and the host may remove any comments related to the event
- Users may search events by browsing, searching for matching words in the title or description, the location using a map display or categories/tags associated with the event

Site Structure



Main page structure consist from header, body and footer.

Header is the same as at the all other pages and include:

- Logo
- Search bar
- Menu, which in turn include:
 - new event. Page for creation a new dinner.
 - My events. List of all dinners made by concrete user. Previous and future.
 - My requests. List of request from other users, who would like to be invited to a planned dinner, if there are any in nearest time.
 - Log in/ my page. First time when user open our service he/she can log in with facebook account and Aalto email.

In the body of the main page there is a list of all planned events and filters for easy searching. Users can search all planned events, use the search bar located in the header or search with filters. From the filter options users can toggle the list view to an alternative map view. Other filters include: time, place, cuisine and gender. By clicking an event an user goes to the page of the chosen event. In the sitemap this page is simply will called an “event”.

The footer of our service is also the same at all pages and include contact information about founders and copyrights.

New event. Page for creation a new dinner. The page consists of a form where you can add

the name of the event, photos, and a description. Users can choose how the dinner is organized: is cooking done together, do people bring their own dishes or has the host prepared everything. Users can write about ingredients they are using in case if somebody has allergies. After submitting a new event it becomes a “event” page and all service users can see it in the list at the main page.

Event. At the event page we can see all information about planning dinner. Also below the description this page include a comments block and small list of requests of participations for this event.

My events. List of all dinners made by the user, both previous and future. On the right side of each event in the list there is a button for opening a modal view with a list of requests from other users, who would like to be invited to a planned dinner. Dinner organiser have function to accept or cancel each request from the list.

My requests. List of all dinners where the user has requested to join. There is a status of acceptance or declination of your requests. From this page the user can also cancel his or her request.

Log in/my page. When an user opens the service the first time he/she can log in with a Facebook account and Aalto email. After logging in the user gets to his profile page where is his photo and name, short introduction and list of hobbies, and lastly past and future dinners with a rating from other users. From this page the user can access the profile settings page.

Settings allow to change photo, name and description of users page.

Maintenance and administration

Since the service contains user generated content there has to be a way to control it. Some users might try to sabotage events, post fake events or just post spam, advertisements etc.

Our service already has an initial no-spam filter, with people only being able to register once with their Aalto email. However, this might not deter all spammers. There are two options for content management

- A) Build a content management system, aka CMS, through which moderator accounts can do administrative actions, such as remove unwanted event postings or ban users, directly from the service UI.
- B) Build an administration panel, where web administrators can remove events and ban users. Instead of being embedded to the service, the panel is on a separate web page.

Option A is technically more difficult, but it allows the possibility to recruit normal users to be moderators. Option B on the other hand is technically more simple, but is not scalable as the service grows.

Because our tight schedule, we have planned to initially build a rather primitive administration panel. These options are non-exclusive, and option B could be later implemented in tandem with the admin panel.

Managing content copyrights

User created content always has a risk to infringe copyright. However, since our service provides no monetization for users, nor can you post any other content that short amounts of texts and pictures, users don't have a motivation to infringe copyrights on purpose.

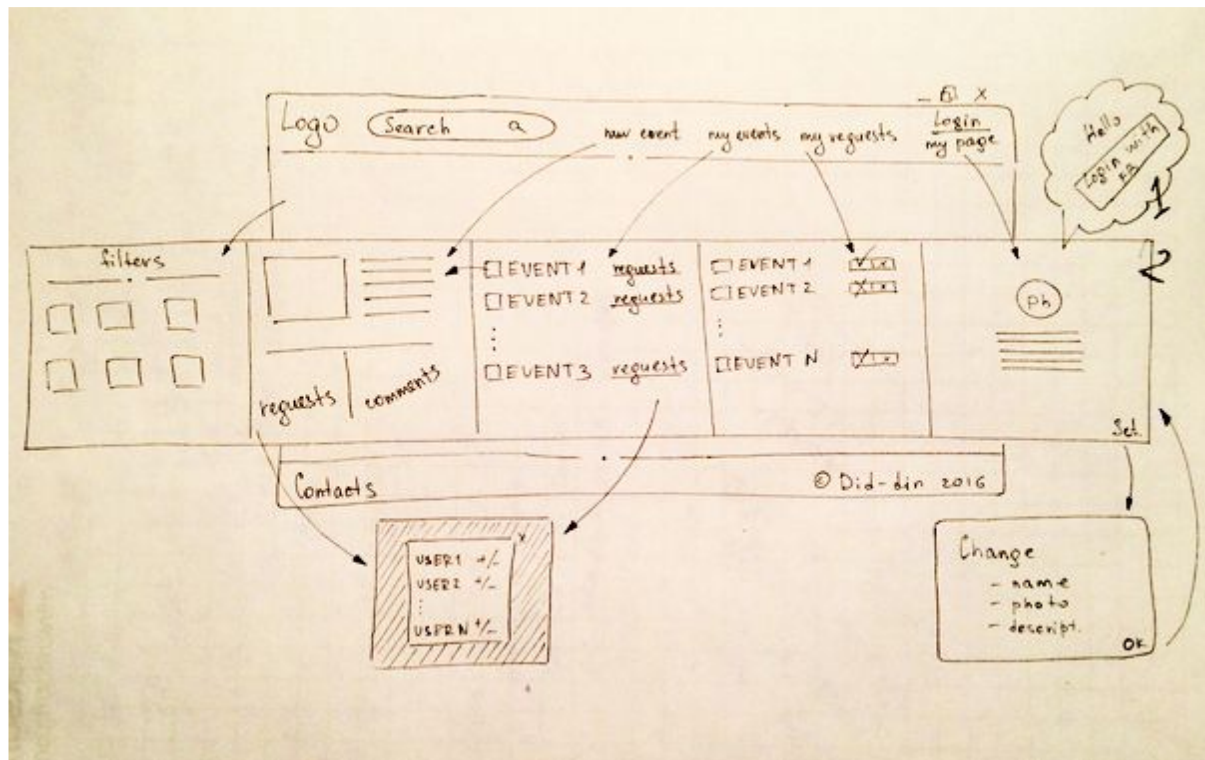
One possible copyright problem could be users posting unlicensed photos to demonstrate the "culinary vision" of their event. This could be lessened by informing users that they must oblige with copyright law and moderating any infringements. The service should be built so that it incentivises using genuine pictures taken by the users themselves.

An alternative scenario is that people would use the service for unintended purposes, such as file or image sharing. This is unlikely, as free file sharing services are ubiquitous. However, by limiting the possibilities to upload content (to certain file types and sizes for example) and making all uploads public, people are disincentivized to do any illegal activities.

In the service itself, we only use images and icons licensed with open or royalty-free licenses such as Creative Commons.

User interface

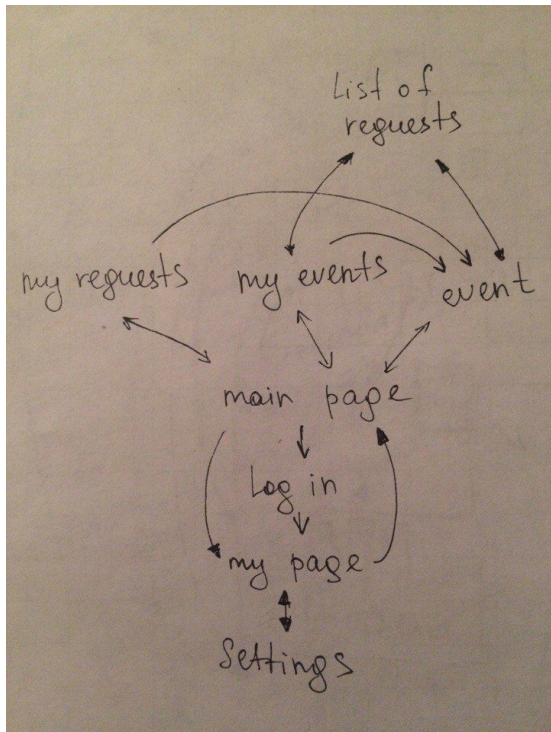
Sketches and descriptions of the UI



As discussed before, our interface has a header and footer on all pages, and the body changes. From the top menu users can switch between several pages such as event, my events, my requests and my page.

Navigation and Interaction

In the chart the page interactions can be seen. The main menu is fixed at the top of the page. From this menu users can navigate to other pages. Users also can go between these pages, for example from page "my events" to any specific "event". At any time user can go back to the main page with click on the logo. We tried to make any page as accessible as possible. As can be seen, the longest navigation task takes only three steps. Navigation paths are shown below.



Fonts and colors

Color

In many cultures yellow represents sunshine, happiness, and warmth. In the natural world, yellow is the color of sunflowers and daffodils, egg yolks and lemons. We think as the name of our service "Din-din" associate with alarm for dinner, and yellow color which captures our attention more than any other color are good in tandem. Also, yellow animates the imagination, promotes the adoption of new ideas and is associated with something interesting.



A lot of famous companies use yellow color in their design, for example McDonald's and Grizzly Bar.



1. http://grizzly-rus.ru/images/logo/grizzly_logo.svg
2. http://vignette4.wikia.nocookie.net/logopedia/images/f/f2/McDonald's_1968.png/revision/latest?cb=20150808083001

The colors for their restaurants McDonald's has chosen not by chance. The combination of yellow and red, and motivates to action attracts even appetizing. Logo is aimed directly at the audience, with the aim of capturing and concentrating its attention on their brands. We decided to take the example from successful companies and implement yellow color into our project.

Colors of our project

	HTML #f1d032 RGB 241, 208, 50 CMYK 0, 13, 75, 6
	HTML #5D5D5D RGB 93, 93, 93 CMYK 0, 0, 0, 64

Font

Almost all the service content generated by users, so there is a risk that the description can be overloaded with information. To get readable text we will try to make the prepared form which user can use for creating an event and describing a dinner. Font impacts on how the user perceive the content.

Depending on the objectives of the project, you can choose a classic font or design font. Our aim is to get the harmonious combination of all elements, because our service is new. It should not shock and be remembered with unusual elements. First of all it must be comfortable.

For the font we have chosen Roboto. The Roboto sans-serif font is a widely used, readable and visually pleasurable font.

Roboto Mono

Glyph

Rr

Thin
Thin Italic
Light
Light Italic
Regular
Regular Italic
Medium
Medium Italic
Bold
Bold Italic

Technologies

Our technology choices reflect three criteria. First, they must be easy to develop, as the course schedule is quite tight. Secondly, this project is a learning opportunity and we want to learn new technologies. Lastly we want to steer towards state of the art and production capable technologies. Even though this project might not need any intensively scalable solutions, we want to aim to a high quality outcome.

Technical outline

Our projects technical stack has been ambitious from the beginning. We want to be able to create a clean and modern feeling service. In practice this means that we have to use some web design patterns that are commonly used in every quality web service.

1. **Single page application.** We want to have a responsive feeling platform. Page loads take time and distract from the usage of the service. Our initial goal is to create a single page application to provide a smooth user experience.
2. **Modern graphical design.** The truth is, most pages look quite similar and there's is a reason - people find it easier to use new services when they are similar to old ones. In our case we want to leverage this to reuse and modify popular front end libraries such as bootstrap. However, creating a sleek modern graphical design is never easy, even if following existing examples.
3. **Respect users.** We don't want to create just another school project. Our team is hoping to create something that could be used, will be used, or at the very least is a gemstone in our future portfolios.

We were able to construct a single page application using modern tools mentioned in this document. However, due to time constraints the UI design was only implemented in a static prototype website and the final implementation didn't use all the designed styles and animation.

Back end

Our team discussed several different technology stacks that can be seen in table 1. We started with our back end. We immediately ruled out java based or other more traditional backends as sub-optimal and cumbersome to develop. This left us a choice between Node and Ruby on rails. Both are used widely in the industry, and even though Ruby on rails is more popular overall, Node has gained a lot of traction in the last few years. Both have good package repositories, and are focused on easy development. Ruby has been criticized for its forceful design layout, which might not fit every architectural design. Because of this, and the overall current popularity dipping slightly in the favor of Node we chose to go with the latter option.

Front end

Front end technologies have a ton of variety and multiple options. As such it is necessarily not easy to pick one existing solution. Nowadays many web services use React, a library

built by Facebook. React is different from our two other main considerations, Sail.js and Angular.js, because it doesn't provide the entire architecture but rather it is only used to construct the view layer. This gives it some advantages, mainly that it is slightly simpler and lighter. Since our stack already included Node, a more flexible back end platform, we had the freedom to choose this choice, as any features that would be missing could be implemented in Node with a open source package from node package managers over 350,000 addons². React also has a slightly different architectural approach than the other libraries and it is especially good with one page applications, because of its virtual DOM technology.

Table 1. Different options for our tech stack

Stack option	Front end	Back end	Database
Final choice	React	Node.js	PostgreSQL
Close second	Sail.js (with React or Angular.js)	Node.js	Sail.js integration with MongoDB or PostgreSQL
Ruby alternative	React	Ruby on rails	SQLite

Dynamic functionalities

Based on the [User test](#) and suggestions from the users we decided to implement next functions:

- Capability to choose more drinks in the filters
- Displaying of the events by proximity map
- Capability to set a price for the dinner.

The two latter functionalities weren't implemented in the technical demo due to time limitations, but can be seen in the latest UI drafts. Similarly, there were shortcoming in other functionalities as well. For example, initially we started implementing the facebook login through Firebase. However, this development process lead to a dead end as we faces several problems with the Firebase platform, which in itself was only a middleman anyway. Because of this, it might have been better to implement the authentication directly into our own backend with something like Passport.js.

As long as we strive for the one page application type service our web page will be fully dynamic. This means that every click will make the page dynamically load new content. Our

² The npm Blog, "Hello, yarn!" <http://blog.npmjs.org/post/151660845210/hello-yarn> [Accessed on 13.10.2016]

web page also has user generated content - which means that most displayed material needs to be dynamically created.

To give a few examples, some examples of dynamic content is categorized in table 2. We have three major contributions to the dynamic nature of our website: user generated content, automated filtering and third party integrations. A more detailed description of the dynamic architecture and database structure is given in the next chapter.

Table 2. Example of different sources of dynamic content

Content	Data source	Stored in
Events	User	Database
Event suggestions	Data filtered based on time, user location, etc.	Databse & realtime variables
User profiles	Facebook API	Third party servers

Back end & Database

The back end system handles retrieving the data from the database for display and saving the data when a user makes changes. The back end was implemented using node.js and PostgreSQL database server. The back end is accessed by the front end application through RESTful API.

RESTful API is a method of accessing the storage, where the data is retrieved, saved, updated and deleted by accessing certain URLs which describe the entity acted upon, using standard HTTP GET, POST, PUT and DELETE methods. The data is formatted using JSON syntax. For example:

- `GET /users/24` gets the user with the id 24
- `GET /events/` gets the list of all events
- `POST /events/` creates a new event with attached JSON data
- `PUT /events/456` updates the event 456 with attached JSON data
- `DELETE /comments/567` deletes comment with id 567

Much of the functionality in the back end was implemented by using NodeJS modules as following:

- Pg for PostgreSQL connection
- Knex.js for database query construction (platform independent)
- Restify for RESTful API implementation
- Passport.js for authentication and access control

Database structure

Table **Users**. This table is used to store user account entities.

- id: unique ID (integer not null)
- email: user's email address (varchar (255))
- password: password hash (varchar (255))
- displayName: user's displayed name in the service (varchar (255))
- description: short description of the user (varchar(2048))
- facebookURL: URL of user's profile page in Facebook (varchar (255))
- profilePictureURL: URL of profile picture (varchar (255))
- createTime: when the account was created (timestamp)
- updateTime: when the account was last updated (timestamp)
- deleted: is the user deleted (boolean, default false)
- lastLoginTime: when the user has logged in last time (timestamp)

Table **Events**. This table is used to store dinner events.

- id: unique ID (integer not null)
- hostId: reference to user entity who's hosting the event (integer references Users:id)
- title: title of the event (varchar (255) not null)
- description: long description of the event (varchar(2048))
- excerpt: short description of the event (varchar (255))
- eventTime: time of the event (timestamp)
- address: address where the event is held (varchar(512))
- location: coordinates of the location of the event for map display (varchar(255))
- imageURL: URL of the event image (varchar(255))
- **guestAmount: how many guests the host wants to have**
- createTime: when the event was created (timestamp)
- updateTime: when the event was last updated (timestamp)
- deleted: is the event deleted (boolean, default false)

Table **Comments**. This table is used to store comments related to events.

- id: unique ID (integer not null)
- title: title of the comment (varchar(255))
- comment: the comment text (varchar(2048))
- createTime: when the comment was created (timestamp)
- userId: reference to user entity who posted the comment (integer references Users:id)
- eventId: reference to the event entity (integer references Events:id)
- hidden: is the comment hidden (boolean, default false)
- deleted: is the comment deleted (boolean, default false)

Table **Requests**. This table is used to link users to the events and describe participation status.

- id: unique ID (integer not null)
- userId: reference to user entity who has requested participation to the event (integer references Users:id)

- eventId: reference to the event (integer references Entities:id)
- message: user's message to event host (varchar(255))
- pending: whether the request is pending and has not been approved or declined (boolean, default true)
- approved: whether the request has been approved (boolean, default false)
- createTime: when the request was created (timestamp)
- updateTime: when the request was reacted upon (timestamp)

Table **Ratings**. This table is used to rate users.

- id: unique ID (integer not null)
- userId: reference to user who did the rating (integer references Users:id)
- targetId: reference to user who is being rated (integer references Users:id)
- type: positive or negative rating (enum: 'positive', 'negative')
- createTime: when the rating was created (timestamp)

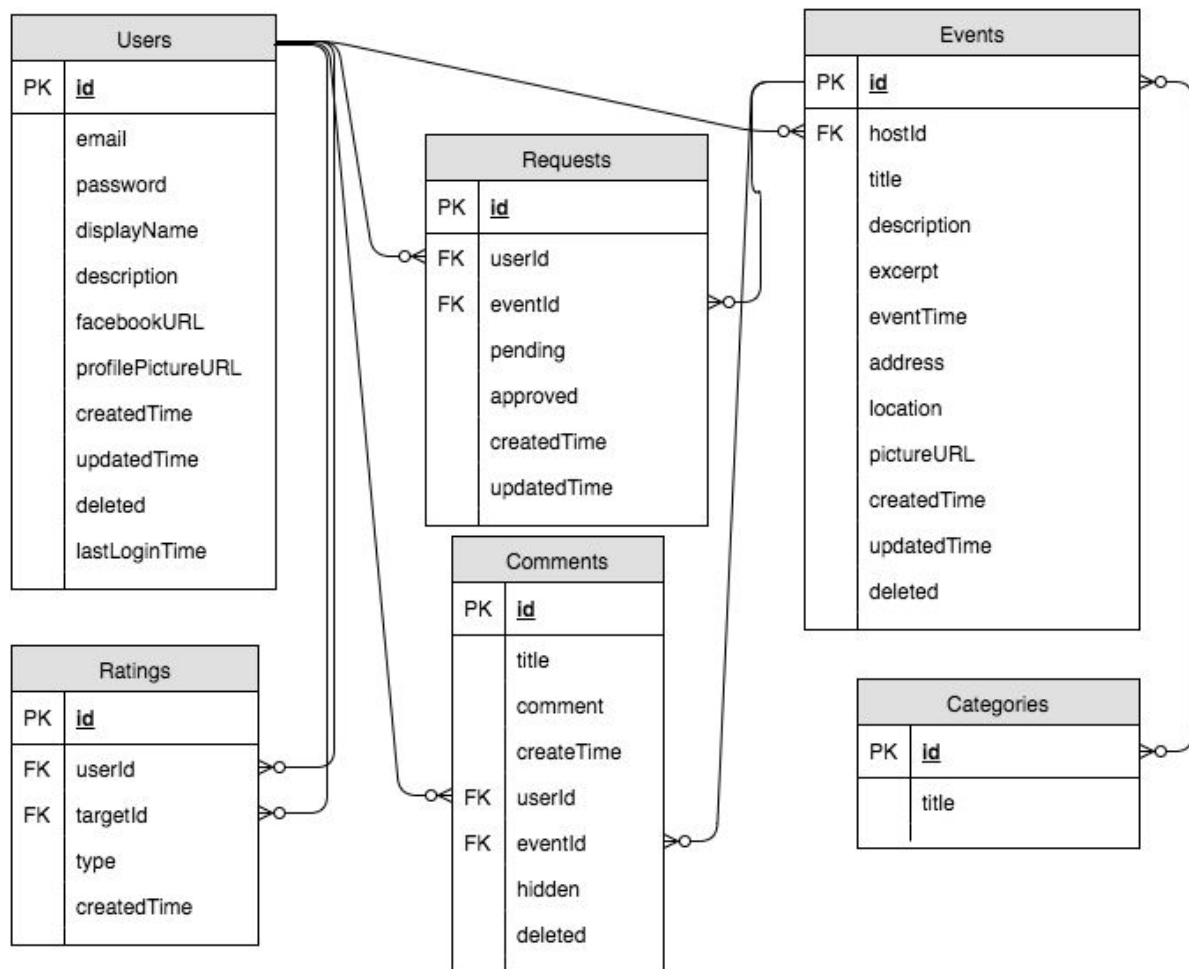
Table **Categories**. This table is used store event categories.

- id: unique ID (integer not null)
- title: category title (varchar(255) not null)

Table **Categories_Events**. This table is used to link categories to the events (many-to-many relationship).

- id: unique ID (integer not null)
- categoryId: reference to the category (integer references Categories:id)
- eventId: reference to the event (integer references Events:id)

The timestamps are automatically updated in the database. When creating records, the "createTime" timestamps have default values for current time. For update timestamps, there are triggers which change the "updatedtime" timestamp upon SQL UPDATE commands.



An entity-relationship (ER) diagram of the database

Security

The service needs to have good security. Because we are dealing with real people, and pairing them with other people, trust is a key issue. The security concerns can be divided into two main categories. First our service deals with people and their personal information (such as real name and pictures). This means that we should take privacy into account. Secondly, we want to target students specifically. This means we need to implement policies that prevent outsiders from snooping around and “crashing” events.

The first problem is a more classical web service problem. Securing information on a web server needs thorough preparations all the way from configuring the actual web server to coding a reliable and secure web application and having secure administration features which unauthorized users cannot access. Because information security is such a deep and well studied field this project does not aim to do anything beyond typical safety measures in that regard, e.g. access control for retrieving and updating information, secure password hashes, input data filtering and validation etc.

The second issue is more about procedures and verification. It is more specific to our service - how do we verify that someone is an Aalto student? One option is to add an extra

registration step where you need to confirm that you have an working Aalto email. However, this makes registration harder, which can immediately turn away some users. As many modern services opt for “soft registration” where your account is immediately created and no email is needed to start using a service, it is hard to balance between security and usability.

Because our Facebook login wasn't completed our project does not have a complete login system. Due to this there are no security features that require login features, even though our architecture supports this and could be easily extended to a version with higher security and authentication levels.

Project Design

Tasks

Task	Priority (1=high, 3=low)	Estimated amount of work (hours)	People responsible
Survey design	1	4h	All
Survey analysis	1	3h	Anastasiia
UI draft	2	6h	Irina
UI Design	1	20h	Irina, all
Competitor analysis	2	4h	Anastasiia
Personas	1	4h	Anastasiia
Interviews	2	6h	All
Requirements design	1	4h	All
Technical design	1	6h	Aleksi, Timo
Database design	1	3h	Timo
Prototype implementation	1	10h	Aleksi, Timo
Server environment	1	3h	Timo
Prototype usability testing	1	6h	All
Basic back end functionality	1	24h	Aleksi, Timo
Search function	3	4h	Aleksi, Timo
Front end implementation	1	30h	Aleksi, Timo, Irina

Facebook login integration	2	6h	Aleksi, Timo
Google maps integration	2	4h	Aleksi, Timo

Project structure

	Week												
	38	39	40	41	42	43	44	45	46	47	48	49	50
Survey													
UI Design													
Interviews													
Competitor analysis													
Requirements													
Prototype implementation													
Prototype testing													
Final implementation													
Documentation													

Initial survey

Initial user data will be collected using an electronic questionnaire. The questionnaire will be sent to target group students. The questions include basic background data from the user group as well as opinions related to dinner organizing. Gathered data will be used to construct personas.

Interviews

User data collection is also done by interviewing potential users of the service. The questions are partly similar to the ones in the questionnaire, but with an emphasis on actual use cases and context of use – where and how the service will be used.

Competitor analysis / benchmarking

In this phase we will pick a few similar websites and conduct a formal analysis regarding what features do they have, what is especially good in the service and what limitations do they have.

Requirements

Requirements will be set based on the data gathered in previous phases. Requirements are formulated as a list, where each requirement is described in detail.

User Interface Design

Interface design changes

At the beginning phase of prototyping our application was looking as it performed in an [Appendix 2. Early prototype screenshots](#). During the coding phase our design has been changing. We followed the target to make our service maximum clear and possibly one page. Therefore we made 3 main design changes: Header visual appearance, event list displaying, page interaction of the service.

1) Header visual appearance

Firstly, we placed search function to the top menu and made it transparent to not attract much attention. Secondly we change the banner and made logo center aligned.

2) Event list displaying

We escaped from table event displaying on the first page and separate info page for each created event and decided to make separate blocks for every event goes one after one. They can be organized by different settings from the user such as time, place, cuisine and other selectors from the filters. This type of event displaying gives us the possibility to escape from the multi page service, because new location of content allows to specify all necessary information about every event in each respective block designated for the event and don't create separate page for it. We also foresee the different color marking button functionality on the event blocks for events creator and other service participants.

- For user creator his event block has light gray color, buttons "edit" and "manage the requests". In its turn the last button "manage the requests" allows to see the list of request (with links to a personal facebook pages), accept/decline the request or change the decision about previous requests. The registration to the service makes with Facebook account, therefore the user can be transferred to the personal facebook page for getting more information about personality who sent a request.

- The events created by other users has wight color and the user has functions only sent/decline the request to attend the dinner and to see the list of other requests without possibility to manage this list, just to be informed about other dinner participants.

3) Interaction between interface pages

With the new specifics of events displaying we decreased the number of pages on the service. Also we made the windows of requests list as a popup window which is also declined the necessity to create an extra pages for this sections.

For more user friendly interface we initially had the buttons “my events” and “my requests” in the top menu. Supposed that this buttons transferred the user to the separated pages with only events created by this user and requests made by him respectively. But since we don't have the separate pages for these two sections, we setted the dropdown list with events/requests which appears after the click on the respective button. And every element in this opened list has the anchor link to the event on the main page.

There is how we went out from the multi page service to a one page.

The picture with marks of the changes you can find in an [Appendix 3](#) and the screenshots of the final visual appearance of the service you can find in an [Appendix 4](#).

Designing the user interface will begin early on in the project. First, a draft of the user interface is constructed, and the design will be improved during the course of the project as more information is gathered about the users, their needs and later in prototype testing. At first, user interface is drafted using simple tools, and functionality will be added later on.

Functional prototype implementation

Functional prototype implementation will start after initial user data has been gathered and user interface design is in such phase that all views have been defined and basic functionality has been described. Functional prototype is a fully functional web app which runs in the browser, with all the views and functionalities in place. All the content does not have to be ready and graphic design doesn't have to be final, as long as the prototype is usable.

Functional prototype user testing

During the Demo gala we not only presented and explained our service but also conducted the user tests. We asked visitors try to use our service and give us any feedback. The user tests showed that probably every user can easy understand the idea and functionality of the service, that the users like the design. Nevertheless we received some suggestions for the improvements. Among them:

1. Make the the search function more understandable. The area for text input and extra button for start searching entered the user confusion.
2. Button with popup window about requests doesn't look like button.
3. Work more with filters, make more sections for the drinks and foresee the search with distance selector.
4. Make the opportunity of setting the price for the dinner, make the service more commercial.

We created the typical [scenario](#) of user test and showed about how some of the users from every our user group could use the service. The collected recommendations from the user test you can see below:

Do you like to add something in filter's ? (9 responses)

no
it is good enough
make extra box for drinks and add vino:)
For the map, distance, also some kind of time filtering could be good
more choices meal types
Distance, price, title of number of guests
nope
price?
age

Tell us a few recommendations and comments about the service (9 responses)

nothing
change some float hint into a label
to conduct in-depth interviews with potential users to determine their behavior on the website. possibly, some buttons or functions are not needed, or vice versa must be added:)
You should try to allow people to make money, by sharing food
nice service
The search function was a bit unclear
:)
id love to use the service. maybe consider pricing to cut the costs.
looks great, many features already implemented

User testing was be organized using a functional prototype of the service. Multiple testing iterations was made as the design matures. At first, testing method is an informal walkthrough, and at a later stage, formal usability testing was conducted.

Scenario of user test

We created several scenarios for user testing that contain user's primary goals. These scenarios include steps with secondary goals as components of the bigger goals. According to user scenarios 2 main scenarios for testing were described and set to the user testing:

1. On the Din-din website create a new cooking event. Find and view requests for participation and approve them.
2. On the Din-din website find the cooking event that is interesting for you and register. The task in the 2nd scenario includes several sub-tasks:

- To find the event that will be interesting to the user.
- To apply for participation in this event.

Each user can do the tasks differently because everyone has different criteria of liking the event: some users can apply the filters on the web-site that will help them to search for specific cuisine or address (closer to their location) or other features of the event. The others will look at the list of accepted guests of each event and base their preferences according to the availability of interesting people in the guest list.

Final implementation

After the prototype has been thoroughly tested, final implementation will be started. The user interface is constructed based on the prototype, and necessary back-end storage procedures will be implemented. Also external API connections will be implemented at the final phase.

Due to limited time available for a 5 credit course, we had to leave out some of the features from final implementation. For example login using facebook and google accounts had to be left out. We studied the possibilities and implementation should be quite straightforward, but it requires registering and setting up our application and API keys in their services, so we decided to skip that. Google maps API is also quite well documented, and it could be integrated to our service, both for displaying locations on map, but also for inputting the location on the form when creating a new event.

Appendix

Appendix 1: User survey

Social dinner organizer

Our group is implementing a web-service for students and would like to know your opinion.

*Required

A few background questions

Background information will help us decide if we should focus on a certain group of people.

1. What is your gender? *

- Male
- Female

2. What is your age? *

- 0-18
- 19 - 25
- 26 - 30
- 31 - 35
- Other:

3. What is your study status? *

- Bachelor program
- Master program
- Exchange/Erasmus/Double degree

4. What is your nationality? *

- Finnish
- Non-Finnish

5. How long have you lived in Espoo/Helsinki area? *

- < 6 month
- 6 - 12 months
- 1-3 years
- > 3 years
- I am local

Using the service

The following questions are related to our service concept

6. Would you like to meet new people on campus? *

- Yes
- No

7. Would you like to have dinner with another student?

- Yes
- No

8. Would you like to invite people to your place to have dinner?

- Yes
- No

9. Would you like to participate in a home dinner by another student? *

- Yes
- No

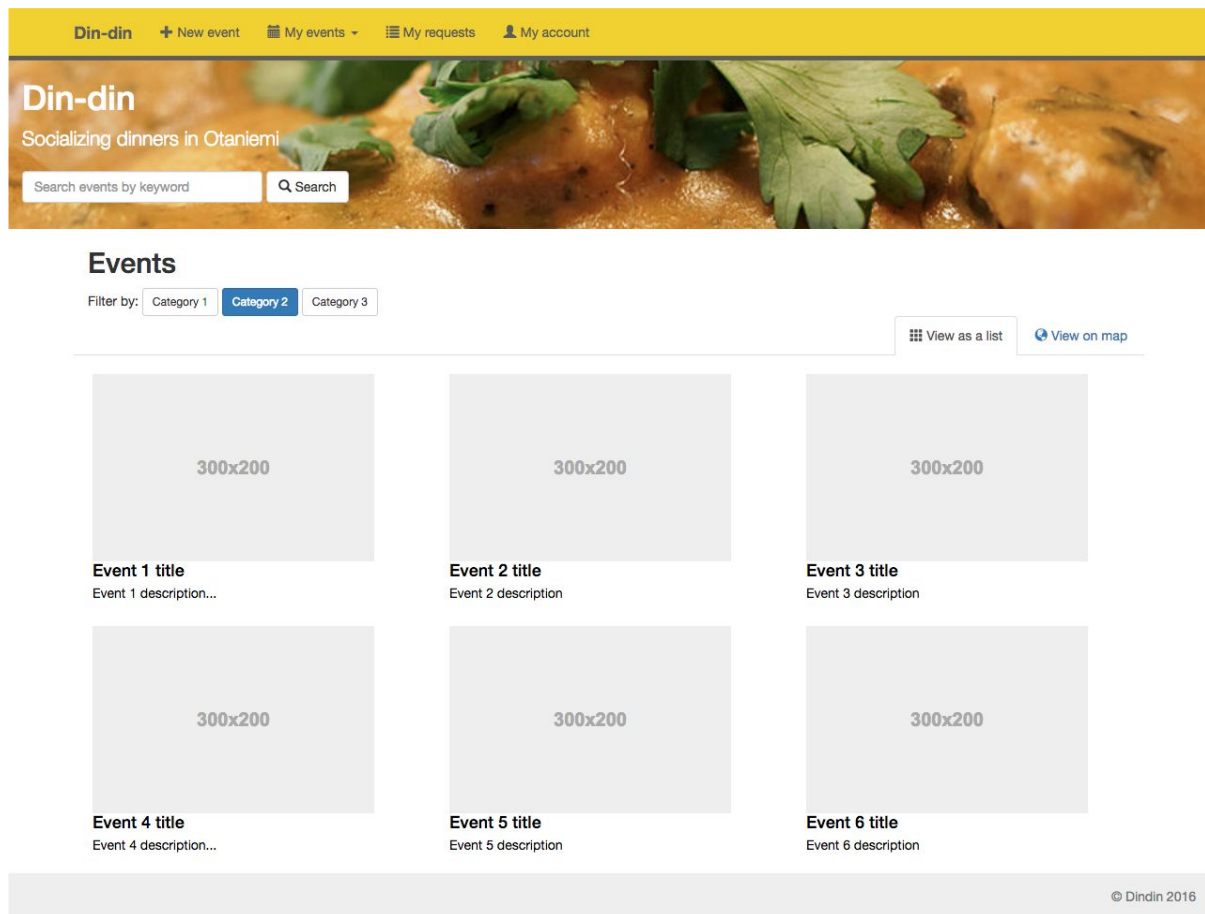
10. Would you like to cook with other students?

- Yes
- No

11. What would you like to know about an event and its organizer before attending it? *

12. What would you like to know about a guest before inviting them to your home? *

Appendix 2. Early prototype screenshots



Front page.

Din-din
+ New event
My events
My requests
My account

Din-din
Socializing dinners in Otaniemi

Search events by keyword
Search

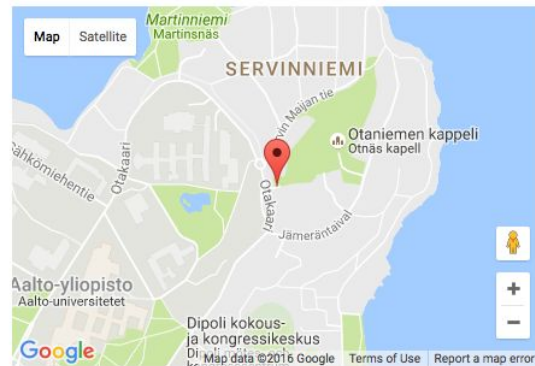
Event title



Description

Description Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

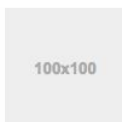
Location


[Request invitation](#)

People attending



User 3



User 4



User 5

Comments



Lorem ipsum

Comment dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua!

User name



Commodo consequat

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

User name

Din-din
+ New event
My events
My requests
My account

Din-din
Socializing dinners in Otaniemi
Search events by keyword
Search

Event title
400x280

Description
Description Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Location
Map
Satellite
Martinniemi
Martinsnas
SERVINNIEMI
Otaniemen kappeli
Otnäs kapell
Jämskänsä
Jämskänsä
Aalto-yliopisto
Aalto-universitetet
Dipoli kokous- ja kongressikeskus
Dipoli
Map data ©2016 Google
Terms of Use
Report a map error

New Requests
User 1
100x100
Short message from user telling about his/her interest.
Accept
Decline
User 2
100x100
Short message from user suggesting how he/she could contribute.
Accept
Decline

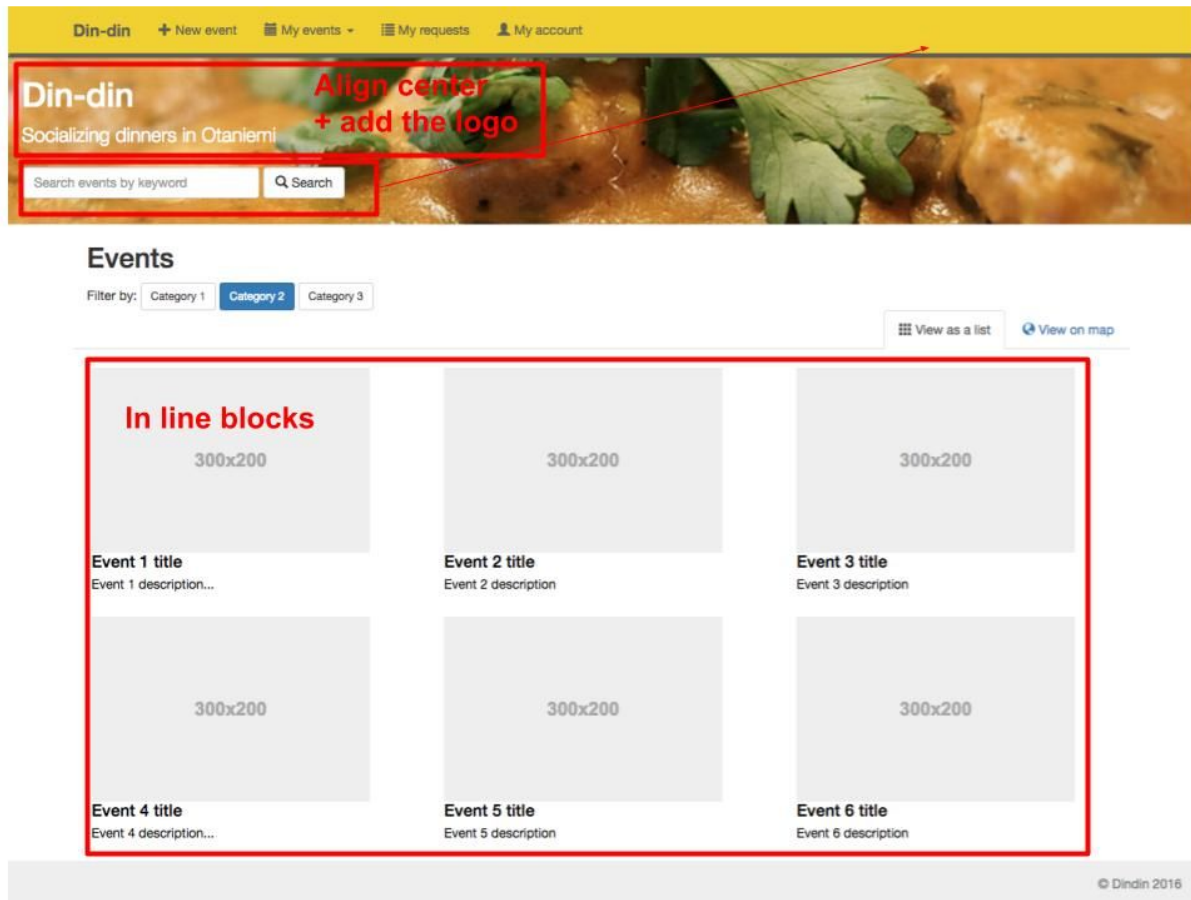
People attending
User 3
100x100
User 4
100x100
User 5
100x100

Comments
Lorem ipsum
60x60
User name
Comment dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua!
Commodo consequat
60x60
User name
Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

© Dindin 2016

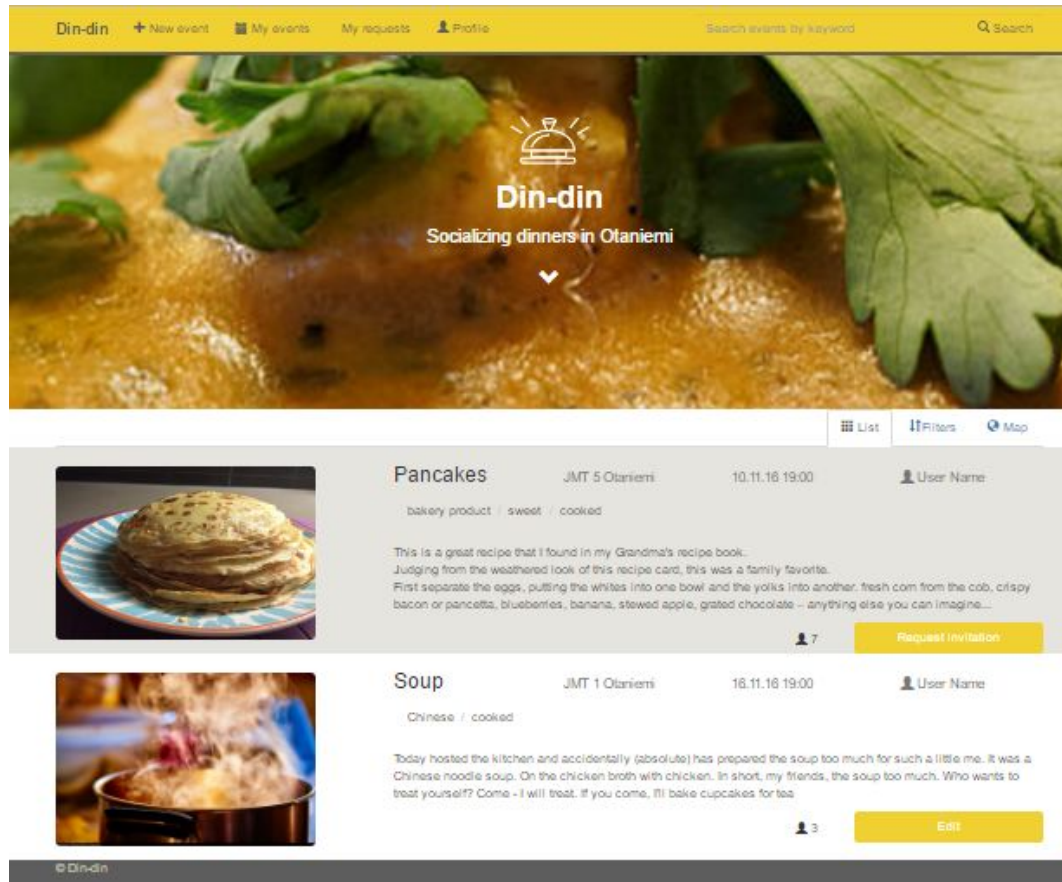
Event page for the host.

Appendix 3. Plan for design changes

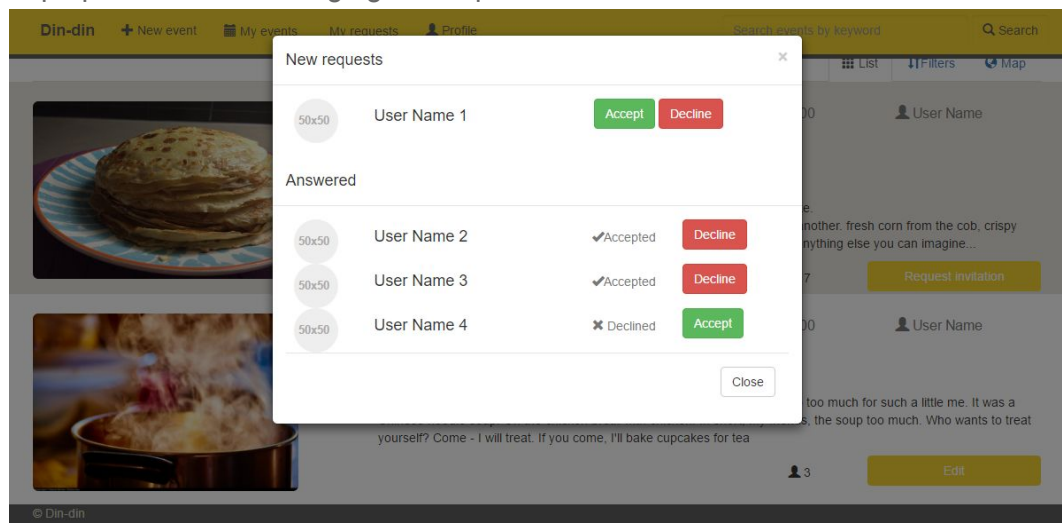


Appendix 4 Final design

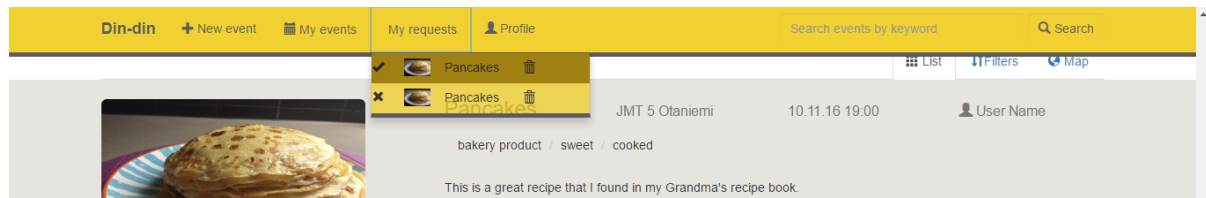
Main page: Search function on the top, centered banner with logo, new event list. Different color of the blocks and different buttons for event creator and simple visitor.



Pop up window for managing the requests.



Dropdown list of “my events” and “my requests” with anchor links



Only one extra page for event creation.

The screenshot shows the Din-din website interface for event creation. The top navigation bar is yellow and contains links for 'Din-din', '+ New event', 'My events', 'My requests', and 'Profile'. A search bar is also present. Below the navigation bar, the event creation form is displayed. The form includes fields for 'Name' (Jane Doe), 'Address' (Otaniemi, JMT 1), 'Photo of your dish' (Выберите файл, Файл не выбран), 'Event time' (19:00), 'Select Cuisine' (Europea), 'Select Kind of dish' (Vegetarian), 'Do you want to cook together?' (Prepared), 'Number of invited guests' (1), and 'Gender preferences' (Male only). A 'Description' field is also present, with a placeholder text: 'write more details about your dinner. you can add recipe, some facts or history of your dish, your expectations about the atmosphere, music preferences, do you need help with cooking and do you need guests something to bring'. A 'Submit' button is located at the bottom right of the form. The footer of the website is dark grey and contains the text '© Din-din'.

Appendix 5. SQL for database creation

```
CREATE TABLE Users (
  id SERIAL PRIMARY KEY, -- unique ID
  email VARCHAR(255), -- user's email address
  password VARCHAR(255), -- password hash
  displayName VARCHAR(255), -- user's displayed name in the service
  description VARCHAR(2048), -- short description of the user
  facebookURL VARCHAR(255), -- URL of user's profile page in Facebook
  profilePictureURL VARCHAR(255), -- URL of profile picture
  createTime TIMESTAMP DEFAULT current_timestamp, -- when the account was
  created
  updateTime TIMESTAMP DEFAULT current_timestamp, -- when the account was last
  updated
  deleted BOOLEAN DEFAULT FALSE, -- is the user deleted (boolean, default
  false)
  lastLoginTime TIMESTAMP -- when the user has logged in last time
);
```

```
COMMENT ON TABLE Users IS 'This table is used to store user account
entities.';
```

```
CREATE TABLE Events (
  id SERIAL PRIMARY KEY, -- unique ID
  hostId INTEGER REFERENCES Users(id), -- reference to user entity who's
  hosting the event
  title VARCHAR(255) NOT NULL, -- title of the event
  description VARCHAR(2048), -- long description of the event
  excerpt VARCHAR(255), -- short description of the event
  eventTime TIMESTAMP, -- time of the event
  address VARCHAR(512), -- address where the event is held
  location VARCHAR(255), -- coordinates of the location of the event for map
  display
  imageURL VARCHAR(255), -- URL of the event image
  guestAmount INTEGER, -- How many guests the host wants to have
  createTime TIMESTAMP DEFAULT current_timestamp, -- when the event was
  created
  updateTime TIMESTAMP DEFAULT current_timestamp, -- when the event was last
  updated
  deleted BOOLEAN DEFAULT FALSE -- is the event deleted
);
```

```
COMMENT ON TABLE Events IS 'This table is used to store dinner events.';
```

```
CREATE TABLE Comments (
  id SERIAL PRIMARY KEY, -- unique ID
  title VARCHAR(255), -- title of the comment
  comment VARCHAR(2048), -- the comment text
```



```

createdTime TIMESTAMP DEFAULT current_timestamp, -- when the comment was
created
userId INTEGER REFERENCES Users(id), -- reference to user entity who posted
the comment
eventId INTEGER REFERENCES Events(id), -- reference to the event entity
hidden BOOLEAN DEFAULT FALSE, -- is the comment hidden
deleted BOOLEAN DEFAULT FALSE -- is the comment deleted
);
COMMENT ON TABLE Comments IS 'This table is used to store comments related to
events.';

```

```

CREATE TABLE Requests (
id SERIAL PRIMARY KEY, -- unique ID
userId INTEGER REFERENCES Users(id), -- reference to user entity who has
requested participation to the event
eventId INTEGER REFERENCES Events(id), -- reference to the event
message VARCHAR(512), -- user's message to event host
pending BOOLEAN DEFAULT TRUE, -- whether the request is pending and has not
been approved or declined (boolean, default true)
approved BOOLEAN DEFAULT FALSE, -- whether the request has been approved
(boolean, default false)
createdTime TIMESTAMP DEFAULT current_timestamp, -- when the request was
created
updatedAtTime TIMESTAMP DEFAULT current_timestamp -- when the request was
reacted upon
);

```

```

COMMENT ON TABLE Requests IS 'This table is used to link users to the events
and describe participation status.';

```

```

CREATE TABLE Ratings (
id SERIAL PRIMARY KEY, -- unique ID
userId INTEGER REFERENCES Users(id), -- reference to user who did the rating
targetId INTEGER REFERENCES Users(id), -- reference to user who is being
rated
type INTEGER, -- positive or negative rating
createdTime TIMESTAMP DEFAULT current_timestamp -- when the rating was
created
);
COMMENT ON TABLE Ratings IS 'This table is used to rate users';

```

```

CREATE TABLE Categories (
id SERIAL PRIMARY KEY, -- unique ID
title VARCHAR(255) -- category title
);
COMMENT ON TABLE Categories IS 'This table is used store event categories.';

```

```

CREATE TABLE Categories_Events (
id SERIAL PRIMARY KEY, -- unique ID
categoryId INTEGER REFERENCES Categories(id), -- reference to the category
eventId INTEGER REFERENCES Events(id) -- reference to the event

```

```
);  
COMMENT ON TABLE Categories_Events IS 'This table is used to link categories  
to the events (many-to-many relationship)';  
  
CREATE FUNCTION update_timestamp() RETURNS trigger AS '  
BEGIN  
NEW.updatedtime = current_timestamp;  
RETURN NEW;  
end;  
' language plpgsql;  
CREATE TRIGGER update_user BEFORE UPDATE ON users  
FOR EACH ROW EXECUTE PROCEDURE update_timestamp();  
CREATE TRIGGER update_event BEFORE UPDATE ON events  
FOR EACH ROW EXECUTE PROCEDURE update_timestamp();  
CREATE TRIGGER update_request BEFORE UPDATE ON requests  
FOR EACH ROW EXECUTE PROCEDURE update_timestamp();  
  
INSERT INTO categories (title) VALUES ('European');  
INSERT INTO categories (title) VALUES ('Asian');  
INSERT INTO categories (title) VALUES ('Other');  
INSERT INTO categories (title) VALUES ('Vegetarian');  
INSERT INTO categories (title) VALUES ('Vegan');  
INSERT INTO categories (title) VALUES ('Cooking together');  
INSERT INTO categories (title) VALUES ('Male only');  
INSERT INTO categories (title) VALUES ('Female only');
```