Bilkent University
Department of Computer Engineering

Senior Design Project
T2308
Perfent

Analysis and Requirement Report

21901631, Beste Güney, beste.guney@ug.bilkent.edu.tr
21802838, Bora Çün, bora.cun@ug.bilkent.edu.tr
21903474, Cemal Faruk Güney, faruk.guney@ug.bilkent.edu.tr
21801831, Çağrı Eren, cagri.eren@ug.bilkent.edu.tr
21902461, Gamze Elif Çenesiz, elif.cenesiz@ug.bilkent.edu.tr
Supervisor: Cevdet Aykanat
Course Instructors: Erhan Dolak, Tağmaç Topal

13.11.2022

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfilment of the requirements of the Senior Design Project course CS491/2.

Contents

Analysis and Requirements Report

Perfent

1 Introduction

Hanging out and going to activities with one's friends, family, or any group can be an excellent way to spend time and create memories. People generally want to attend events together with others whom they wish to spend time with. These events should be interesting to them and that are a reasonable fit for their schedule. For example, if one suggests an event to their friend group that their friends are passionate about and that covers a reasonable free time slot on their friend’s schedule, most people would be happy to join such an event with their friends.

When it comes to organizing, several problems emerge such as none of the group members taking the initiative to organize, finding a proper time available for everyone, the tiresome activity of researching and choosing an event among countless websites, and finding an event that the group members will be interested in. These problems may cause them to waste a huge amount of time scheduling and browsing events on the internet to find an event that will be the best fit for the group. In addition, people can also miss an event that they would otherwise prefer to go to since they were not aware of that event. Perfent aims to bring solutions to these problems and transform the organization process into a more autonomous and personalized experience for groups.

An algorithm will be developed in order to detect the joint available slots of the group member and events that occur at that slot will be suggested to the group taking into consideration their previous choices and preferences. To do so, a recommendation system that uses collaborative filtering will be developed and it will be applied for both groups and individuals separately. Events will be taken from several different websites by web-scraping and will be combined together.

2 Current System

There are other applications that are based on events and event recommendations; some of the most known are Meetup [1] and Eventbrite [2]. Meetup is about communities hosting events in themselves and other users joining these communities and joining the events that are hosted by those communities. Eventbrite is about users browsing local events, creating events, and buying tickets for those events. There are also other similar applications such as AllEvents[3], Unation [4], TickPick [5], Gametime [6],

TicketMaster [7], and StubHub [8] where users can browse events and buy tickets for them although type of the events focused by these applications differ from one another. For example, Gametime [6] focuses on sport game based events while Unation [4] focuses on all types of events.

The fundamental difference between Perfent and these applications is that Perfent focuses on the event experience of groups of people rather than individuals and most of its feature set is around supporting groups having the best experience at easily planning a group outing for an event they will be interested in and attending the event. Although it borrows some best practice features such as event browsing, personal recommendations, group finding and ticket buying (although it is low priority for us) from the applications mentioned above.

Nowadays, when a group of people (family members, friends, acquaintances etc.) are interested in attending an event with their group; they are left with the complex problem of functioning and handling the event process as a group of people. Other applications mentioned above cannot offer much help because inherently the process of going to an event with a group vastly differs compared to going to an event as an individual. When a group of people are involved, the event process includes decision making, communicating the ideas and needs, considering interests of all group members and considering real life constraints of all group members such as time, money, location etc. Different from the other applications in the field Perfent offers solutions for this aspect of the event attending process.

3 Proposed System

3.1 Overview

Perfent will be a web-based application that recommends events to attend to groups of people. In this subsection, an overview of the features are categorized and explained.

3.1.1 Groups

After signing in to Perfent, the users will be able to be parts of one or more groups. The group system will be very similar to that of Whatsapp's group system [9].The users will be able to either create a group or join an existing one with invites. There can be one or more admins that have access to special operations such as sending invites and removing members.

The users are able to join more than one group and they can "switch" between the groups so that they can see events and data that is related to that "current group."

The main reason a group is created is to find the best events to attend to with that group. According to the group's collective interests and their time availability along with some customized parameters that the group sets; a set of events will be presented to each group. Alternatively, the group can browse a list of all events, even if those events are not related to the group. If the group is interested in one of these presented or other events, they can attend that event. Detailed information about choosing an event will be explained in the following subsections.

Other than the already existing events, the group can create custom events that they can go to. In a sense, the group uses Perfent as a meeting planner in this case. Since they can see the free time slots of the group, it is easier to plan an internal event.

As an additional feature, the group members will be able to create "notes" and attach them to events of their choice. The notes are a general purpose feature and the members can use these for whatever they want about the event they are going to. One specific use of these notes is to assign which group members should bring which items that are needed in the event. For example, if a group is going camping, some equipment must be brought and the notes are an easy way of reminding the group members who should bring what.

As another additional feature, each group will have their own private feed where they can post photos, videos, and comments from an event they went to together. This provides a nice way of collecting memories of the times spent together and an incentive to go to more events to make their feed larger.

3.1.2 Schedules

Schedules in Perfent help the group determine their common free times. Each person has a dedicated schedule for each of their groups. Using these individual schedules, Perfent creates a "group schedule." This group schedule shows the times that all the group members are available. The group schedules provide a nice way of visualizing the availability and they are also used to filter the recommended events, so that all the members can go to the recommended events.

As mentioned before, the individuals have a different schedule for each of their groups. The difference in these schedules is based on people having different availability for different groups. For example, a user's lunch break times may be available for a work group, but it cannot be for the groups where the user has to travel a long way.

In the early iterations of Perfent, the users will only be able to create schedules using Google Calendar [10]. In the later stages, Perfent's own schedule interface will be implemented. Alternatively, people will be able to import calendars from different popular third-party services in the later stages.

If a time period in a user's schedule is free, but the user does not want to attend an event at that time; they can set that time period as "busy" directly from Perfent without changing their calendar in a third-party calendar. Similarly, they can set a time period as "available" if that time period is actually not available.

3.1.3 Events and Recommendation

Perfent will gather a variety of events from popular event and ticket sites using web scraping. These events will be processed and categorized automatically. Then, based on the groups' and individuals' past event preferences and their clickstream data, new events will be recommended to the groups. These recommended events will take the group members' availability, price, distance, age, and similar preferences into account. The system will periodically recommend events to the groups.

The users do not have to wait for the system's recommendations, they can browse all the upcoming events and make suggestions to their groups as well. In Perfent's terminology, this is called "proposing an event." By using various sorting and filtering options, the users can find suitable events easily.

The events that are recommended by the system or proposed by the group members can be viewed in a list. In this list, the group members can indicate their opinions by agreeing, disagreeing, or staying neutral with each event. Perfent assumes that each group already has a platform for verbal discussion and does not complicate the implementation process by including a group chat feature. The agreeing and disagreeing provides a handy visualization for the group members which allows them to understand what the rest of the group wants. Of course, this process does not compel the group to attend that event. This is just for understanding the group's stance on the events.

If there are more than one event that the group converges on, but they can only choose one due to time constraints or other external reasons; the members can call a vote between two or more of these events that the group wants to go to. Using the result of the vote as a kind of a tiebreaker, the group can decide which event to attend. Again, the group does not have to attend the winning event, this is just a tool to ease the decision process.

Using features mentioned above and understanding their stance on attending the event, the group can choose to attend an event. Then, they can mark the event as ”will-be-attended” indicating that they will attend the event. By doing so, the group can use the event attending capabilities of the Perfent.

In the early stages, a ticket buying or event reservation system will not be implemented for events found from the internet. However, to make things easier, Perfent will redirect the users to secure third-party services where they can perform these actions.

The users can add events and event artists to their wishlist. This helps improve the recommendation accuracy and the users can get notified about

the upcoming events and event artists from their wishlists. Another feature that helps the recommendation algorithm and other users is that the users can rate the events. Similar to this, all the events will have an option where the users can choose to see more or less like that event. Aside from the rating data, clickstream data will be collected and processed to get an idea of which events draw the users attention. When recommending, the event ratings will be more dominant in the recommendation choice. However, we expect a scarcity in this type of data and consequently decided to include clickstream data to help.

3.1.4 User Matching

As an additional feature, Perfent helps the users find new people to attend events with. This is an optional feature and all the users are opted out by default. If a user opts in for this feature, Perfent presents a list of other users with similar interests and events so that the user can meet with one or more of these people to attend relevant events. This feature is especially useful for those who want to attend a particular type of event, but have no one they know that wants to go to that type of event. With this feature, they can find people in similar situations.

3.2 Functional Requirements

3.2.1 Event Functionalities

- When a new user joins the system, the system will present an optional questionnaire (normal questions, ask previously joined events, present them with some events, and ask which ones they would attend) to gather user preferences.
- The users can mark the events with “show more/less like this” options to provide feedback to the recommendation algorithm.
- The users can browse all the upcoming and past events with filtering, sorting and search functionalities.
- The users can browse the events suggested to their currently chosen group.
- The users can browse the events suggested to themselves only (independent from any groups).
- The users and groups can add constraints to their suggestion algorithm such as price, age, time, number of minimum available members, etc.
- The users can add events and event artists to their “Wishlist” and their groups’ “Wishlists” to receive notifications and improve suggestion accuracy.
- The users can rate events and post comments about them to improve suggestion accuracy as well as to help other users decide if the event is a recurring event.
- The system will notify the events that are the best fit for the group periodically.

3.2.2 Group Functionalities

- The users can create groups.
- The users can join already existing groups via invitations.
- The group members can view basic information about their groups such as name, members, creation date etc.
- The group members can propose events to their groups.
- The group admins can invite other users to their group.
- The group admin can remove members from their group.
- The group admin can make other group members group admins.
- The groups can view the events they have attended.
- The users can switch between their “group views” so that they can see what events are suggested to their currently chosen group.
- The groups can organize internal group activities.
- Groups can add important notes to the events they plan to attend, recommended events by the system and proposed events by the users.
- The group member can vote among events if the members wish to choose one of many agreed events.
- The group members can leave the groups they want.
- The group members can optionally agree or disagree to events proposed by the group members or recommended by the system.
- After a group has agreed to attend an event on the system, they can create a list of required items that will be brought to the event.
- Groups can assign group members to the items indicating who should bring which item.
- Item bringers will get notifications from the system so that they don't forget the items.
- Groups can have a private group feed where they can share photos and videos.
- The users and groups can mark events as attended.

3.2.3 Schedule Functionalities

- The users and groups can have a schedule view that shows events in the free slots.
- The users can mark the schedule cells as occupied and add activities to their schedules.
- The users can import an already existing schedule from third-party applications.
- The users can synchronize their schedules if the third-party schedule is updated.

- The group members can view the proposed events by other group members and recommended events by the Perfent both on the time slots on the calendar and in a different section (e.g. notifications).
- The users can indicate between which dates they will not be available to join events so as not to disrupt the recommendation algorithm. Between the specified dates, the user will be “invisible” to the system.
- Similarly, the uses can indicate if they are available in a time period even if their third-party schedule says they are busy. These functionalities can be thought of as “overriding” the imported schedules.
- The users can hide the activities in their schedules from other users. If they decide to do so, they will not be able to see the activities of other users.

3.2.4 User Matching Functionalities

- If the user opts in, the system will include the user in a user-matching algorithm. With this functionality, if the user wants to meet other people to go to an event with, the system will help them find those people.
- After attending an event users can evaluate the system-recommended users they have gone to an event with as system feedback, the evaluation will not be shown to any users.
- Users can report the system-recommended users that they have gone to events with.
- Support team can review the reports and take appropriate action.
- Users can view the past users they have attended events with.
- Users can block users from getting recommended to them.

3.2.5 Basic Features

- The users can have a profile page where other users and they can view information about the user such as name, surname, email etc.
- The users can edit some information about them such as email, password etc.
- The user can mute notifications coming from the system.

3.2.6 Other Low-Priority Functionalities

- The users can anonymously post photos and videos from the events they go to so that the viewers can have an idea of what the events are like (They might be curious about what they missed or the event might be a recurring event).
- Groups can have an event feed where they can post pictures from the event. Later these pictures can be shown in the group feed as well.
- The “support staff” can verify event runner accounts and check user reports.

- Event runners can create new events by providing all the necessary event information such as description, price, venue information, etc.
- Event runners can cancel the events they created by notifying the ticket holders and refunding the money.
- The users can buy tickets for the event they are attending as a group/solo (Only applicable to events created in Perfent).
- Event runners can request an “event runner” role from the support staff by uploading evidence to the system that shows that they are capable of organizing events.
- After a group has agreed to attend an event, group members can get recommendations for the least time-consuming ways to reach the event location.
- The users can combine their imported schedules if there are multiple.

3.3 Non-functional Requirements

3.3.1 Maintainability

The application will have the necessary documentation and tools set up to enhance maintainability which is the ease of modifying a component or a system to correct faults and improve performance or other attributes [11]. To satisfy such needs our application will use the following metrics and target a maximum of 5% code duplication threshold, a minimum of 80% unit test coverage, and a maximum cyclomatic complexity of 20 for each unit [12].

3.3.2 Availability

The application will be available for most of the time of its lifetime. Our application will aim for a minimum availability of 99% during its lifetime. Most services on the internet fall between 99% and 100% of availability and our application targets to be like one of those services at the bare minimum [13].

3.3.3 Usability

The user interface of the application should be easy to manage, simple to use, and usable. It will ensure that all of the pages of the user interface can be understood at a reasonable level and traversed in a maximum of 1 minute.

3.3.4 Safety

Any private personal information entered into the system by the user such as interests or addresses will not be disclosed to the public and will be safeguarded by the servers.

Passwords entered into the system will be hashed with effective hashing algorithms that further protect them [14].

The application will have the necessary features to ensure that users are going to hazard-free events with hazard-free users.

3.3.5 Scalability

Our servers should be able to scale when it is necessary and handle the requests incoming from 5,000 concurrent users seamlessly and without any repercussions to the users using the website and the availability of any of Perfent's functionality. It should be able to load-balance the coming traffic when the traffic gets heavy since not managed traffic can cause lags in the system and lag can be a determinant factor in losing a customer [15].

3.3.6 Performance

The application will satisfy the user's waiting time expectations and prevent users from bouncing off our website. The application will target a 2-second loading time threshold with a 6-7% bounce rate for the initial (entry) loading of the website [16]. Then, for each loading of the other pages, it will target the 1-second loading time with a 6-7% bounce rate [16]. Finally, for other actions of the user in the user interface that do not include server interactions, it will target the maximum action time of 100ms.

3.3.7 Portability

The website will also be portable when viewed from devices that are not computers such as mobile devices. All of the features that operate when the website opens from a computer will also operate and will be easy to use when it is opened from a device that is not a computer. This is important because as of August 2022 53.74% of all internet traffic is coming from mobile devices instead of computers [17].

3.4 Pseudo Requirements

- The program will be a web application.
- The application will follow a client-server architecture. While the client side will create a platform for users to interact with the application, the server side will consist of three different modules.
    - The first module will scrap the web and process the data to collect data related to events for the application.
    - The second module will apply machine learning algorithms to create recommendations
    - The third module will be the web server to create the connection to the client side of the application.
- The project will be developed using Git version control system. Github will be used as the version control platform.
- The server side of the application will run on AWS platform.
- As being developed on the web, the application needs to be highly responsive and user friendly. Because of the fact that the application

will have different modules, they need to be integrated and interact with each other fast.

- For user friendliness, the application should have a clear and visually-appealing user interface.
- The web interface should be responsive to the device resolutions. In other words, when users enter the website from their mobile devices or smaller size screens, the application should regulate the interface accordingly.
- The application should be dynamic. To keep track of the changing schedules of the users and the updated events, it needs to fetch data regularly. These data fetching intervals should be determined delicately to avoid overloading the application.
- To test the plausibility of the recommendation system, synthetic data needs to be generated so that even before having registered users in the system, the application behavior can be observed.
- The application should be available in English and Turkish.

3.5 System Models

3.5.1 Scenarios

3.5.1.1 Group Operations

Use case name	Invite Users to Groups
Participating Actor	Group Admin
Flow of Events	1) Group admin opens the group tab that he/she wishes to invite somebody. 2) Group admin enters the email of the person he/she wishes to invite. 3) Group admin clicks the send button to send the invitation.
Entry condition	Group admin needs to be logged in. Group admin needs to be the member of the group that he/she is inviting somebody into.
Exit condition	Group admin sees the invitation sent prompt.

Use case name	User Joins Groups
Participating Actor	User
Flow of Events	1) User opens notifications. 2) User views the invitation requests. 3) User accepts the invitation and joins the invited group.
Entry condition	User needs to be signed in. User must not be the group member of the group he/she accepts to join.
Exit condition	User sees the “joined to the group” prompt.

Use case name	Leaves Groups
Participating Actor	Group Member
Flow of Events	1) Group member opens the page of the group they wish to leave. 2) Group member clicks the leave the group button in the group page.
Entry condition	Group member needs to be signed in. Group member needs to be the group member of the group he/she is going to leave.
Exit condition	Group member sees the “left the group"

	prompt and is redirected to another page.

Use case name	Remove Members From the Group
Participating Actor	Group Admin
Flow of Events	1) Group admin opens the page of the group. 2) Group admin clicks the tab where group members are shown. 3) Group admin clicks the remove member button near the group member that he/she wishes to remove from the group.
Entry condition	Group admin needs to be logged in. Group admin needs to be the admin of the group he/she removes the member.
Exit condition	Group admin sees the group member is removed from the group prompt.

Use case name	User Creates Groups
Participating Actor	User
Flow of Events	1) A pop-up for group creation is displayed. 2) User enters information for the group such as group name. 3) Users can invite other users to the group.

	4) User clicks the create button.
Entry condition	User needs to be logged in.
Exit condition	User sees the group created prompt.

Use case name	Make Members Group Admin
Participating Actor	Group Admin
Flow of Events	1) Group admin opens the page of the group. 2) Group admin clicks the tab where group members are shown. 3) Group admin clicks the make group admin button near to the group member that he/she wishes to make the group admin.
Entry condition	Group admin needs to be logged in. Group admin needs to be the admin of the group he/she makes the members admin.
Exit condition	Group admin sees the group member is given the group admin role prompt.

Use case name	View Basic Group Information (Name, creation date, members etc.)
Participating Actor	Group Member

Flow of Events	1) Group member opens the group's page. 2) Group member views the group information.
Entry condition	Group member needs to be logged in. Group member needs to be the member of the group that he/she is viewing.
Exit condition	Group member closes the group's page.

Use case name	Propose an Event to the Group
Participating Actor	Group Member
Flow of Events	1) Group member opens the page of the event that he/she wants to propose to his/her group. 2) Group member clicks the "propose event" button. 3) Group member chooses the group that he/she will propose the event to. 4) Group member clicks the submit button for a proposition to appear for the group.
Entry condition	Group member needs to be logged in. Group member needs to be the member of the group that he/she is proposing an event to.
Exit condition	Group member sees the "event proposed" prompt.

Use case name	View Proposed Events

Participating Actor	Group Member
Flow of Events	1) Group member opens the tab that shows proposed events of the group. 2) Group member views the proposed events by the other group members.
Entry condition	Group member needs to be logged in. Group member needs to open the page of a group who he/she is a member of.
Exit condition	Group member views the proposed events and closes the tab that shows the proposed events.

Use case name	Edit Suggestion Preferences for Groups and User
Participating Actor	User
Flow of Events	1) User enters the profile page or the group's recommendations page. 2) Under the recommendations tab, the user changes the constraints (price, time interval, distance, rating etc..) 3) User presses the save button at the end of the page.
Entry condition	User needs to be registered and signed in.
Exit condition	User clicks on the save button and goes back to the profile page.

Use case name	Add Notes to the Proposed/Recommended/Will be Attended Events
Participating Actor	Group Member
Flow of Events	1) Group member clicks the proposed, recommended or will be attended events of the group to open its details. 2) Group the member clicks the add notes button. 3) Group member writes the notes he/she wishes to write. 4) The group member clicks the save button to save the written notes.
Entry condition	The group member needs to be signed in. Group member needs to be the member of the group that he/she is writing notes on the event of.
Exit condition	Group member sees the note saved prompt.

Use case name	Vote Among Agreed Events
Participating Actor	Group Member
Flow of Events	1) The group members or the application suggests events to the group. 2) The group member enters the group page. 3) The Group Member views the suggestions. 4) For each suggestion, group member click on either agree or disagree button. 5) The system confirms if the group member is certain from its vote. 6) The system accepts the vote after

	confirmation; if it does not the group member votes again.
Entry condition	User needs to be signed in and a member of a group.
Exit condition	The user vote is taken by the system.

Use case name	View Attended Events
Participating Actor	Group Member
Flow of Events	1) Group member opens the group's page. 2) The group member clicks the attended events tab. 3) The group member views the attended events.
Entry condition	The group member needs to be logged in. Group member needs to be the member of the group that he/she is viewing the attended events of.
Exit condition	The group member closes the tab that shows attended events.

Use case name	Add Event Artists to Group Wishlist
Participating Actor	Group Member
Flow of Events	1) The group member clicks the event page of the events when he/she is browsing proposed, recommended events or events at the browse page.

	2) The group member clicks the add to wishlist button that is near the event artist's name.
Entry condition	The group member needs to be signed in. Group member needs to be the member of the group that he/she is modifying the wishlist of.
Exit condition	The group member sees the event artist added to the wishlist prompt.

Use case name	BrowseEvents
Participating Actor	User
Flow of Events	1) User enters the events view tab or clicks on the details for the event suggested to them. 2) After viewing, the user switches back to a different page.
Entry condition	User needs to be signed in.
Exit condition	User switches to a different tab.

Use case name	Add Events to Group Wishlist
Participating Actor	Group Member
Flow of Events	1) The group member clicks the event page of the events when he/she is browsing proposed, recommended events or events at the browse page. 2) Group member clicks the add to

	wishlist button that is near the event's name.
Entry condition	The group member needs to be signed in. Group member needs to be the member of the group that he/she is modifying the wishlist of.
Exit condition	The group member sees the event added to the wishlist prompt.

Use case name	View the Group Feed
Participating Actor	Group Member
Flow of Events	1) The group member clicks the group feed's tab button. 2) The group member views the feed of the group.
Entry condition	The group member needs to be logged in. Group member needs to be the member of the group that he/she is viewing the feed of.
Exit condition	The group member closes the group feed.

Use case name	Post on Group Feed
Participating Actor	Group Member
Flow of Events	1) The group member clicks the group feed's tab button. 2) The group member clicks the button that lets them pick a photo or a video

	from their device. 3) The group member chooses a photo or a video from their device. 4) The group member clicks the send button sending their chosen photo or a video.
Entry condition	The group member needs to be signed in. The group member needs to be the member of the group that he/she is sending a video or a photo.
Exit condition	Group member sees their photo or a video posted on group

Use case name	Agree/Disagree Proposed and Recommended Events
Participating Actor	Group Member
Flow of Events	1) The group member clicks the tab button that shows the proposed and recommended events. 2) The group member clicks the agree or disagree button that is shown on each proposed and recommended event.
Entry condition	The group member needs to be signed in. Group member needs to be the member of the group that he/she agrees or disagrees with the proposed and recommended event.
Exit condition	The group member sees the confirmation prompt of their agreement or disagreement of the proposed and recommended event.

Use case name	Receive and View Notifications for Best Fit Group Event Recommendations Periodically
Participating Actor	Group Member
Flow of Events	1) The group member opens the notification channel where notifications arrive. 2) The group member views the notification.
Entry condition	The group member needs to be the member of the group that the recommendation is arriving to. The group member needs to allow notifications from the Perfent.
Exit condition	Group member closes the notification after viewing it.

Use case name	Decide on Attending an Event
Participating Actor	Group Member
Flow of Events	1) The group member opens the event's page. 2) The group member clicks the attend button indicating that their group will attend that event.
Entry condition	The group member needs to be signed in. The group member needs to be the member of the group that he/she is accepting for his/her group that they will attend the event.
Exit condition	Group member sees the "event will be attended" prompt.

Use case name	Organize Internal Group Activities
Participating Actor	Group Member
Flow of Events	1) The group member opens the group's page. 2) The group member clicks the organize internal activity button. 3) The group member enters the necessary information for creating an activity (name, time etc.). 4) The group member clicks the submit button to create the activity.
Entry condition	The group member needs to be logged in. The group member needs to be the member of the group that he/she is creating an activity in.
Exit condition	The group member sees the activity created prompt.

3.5.1.2 Event Operations

Use case name	Search Events
Participating Actor	User
Flow of Events	1) User clicks on the search bar 2) User types keyword(s) to find the desired event(s) 3) The desired event(s) is displayed on the page
Entry condition	User opens "Events" page

Exit condition	User leaves the page

Use case name	Filter Browsed Events
Participating Actor	User
Flow of Events	1) User clicks on the filter button 2) User selects the filter options of price, time interval, distance, rating etc. 3) User clicks the apply button. 4) The filtered events are displayed on the page.
Entry condition	User opens the “Events” page.
Exit condition	User browses the filtered events.

Use case name	Sorts Browsed Events
Participating Actor	User
Flow of Events	1) User clicks on the sort button. 2) User selects the sort option from price, time interval, distance, rating etc. 3) User selects between ascending and descending. 4) User clicks the apply button. 5) The events are displayed according to

	the new sorting logic
Entry condition	User opens the “Events” page.
Exit condition	User browses the sorted events.

Use case name	BrowseEvents
Participating Actor	User
Flow of Events	1) User enters the events view tab or clicks on the details for the event suggested to them.
Entry condition	User opens the “Events” page.
Exit condition	User switches to a different tab.

Use case name	User Adds Event Artists to Wishlist
Participating Actor	User
Flow of Events	1) User clicks on the add to wishlist button near the name of the event artist 2) The event artists is added to user’s wishlist 3) User receives notifications for event artist’s new events

Entry condition	User is viewing an event that he/she wishes to enter in the future.
Exit condition	User receives the “event artist added to the wishlist” prompt.

Use case name	Add Events to Wishlist
Participating Actor	User
Flow of Events	1) User clicks the event page of the events when he/she is browsing proposed, recommended events or events at the browse page. 2) User clicks the add to wishlist button that is near the event’s name.
Entry condition	User needs to be signed in.
Exit condition	User sees the event added to the wishlist prompt.

Use case name	View Notifications for the Event Wishlist
Participating Actor	User
Flow of Events	1) An event at the wishlist’s date is approaching or an event artist at the wishlist posts a new event. 2) Perfent sends the notification to the user. 3) User views the notification.
Entry condition	The user needs to have the event or event artist that notification is sent for in their

	wishlist. The user needs to allow notifications.
Exit condition	User closes the notification.

Use case name	Fill Event Questionnaire
Participating Actor	User
Flow of Events	1) User joins the Perfent application. 2) Perfent offers users to fill an event questionnaire to improve recommendations. or 1) User opens the profile page. 2) User clicks the fill event questionnaire button. 3) User fills questions asked in the questionnaire (tick the interested event types, would you join this event questions). 4) User submits the questionnaire.
Entry condition	User needs to be signed in.
Exit condition	User receives the questionnaire submitted prompt.

Use case name	Mark Events as Show More Similar Events or Show Less Similar Event
Participating Actor	User
Flow of Events	1) User sees an event while using Perfent. 2) User marks the event as “show more

	similar events” or “show less similar events”.
Entry condition	User needs to be signed in.
Exit condition	User receives the event marked prompt.

Use case name	Browse Group Recommended Events
Participating Actor	Group Member
Flow of Events	1) Group member opens the group’s page. 2) The group member opens the recommended events page. 3) The group member views and browses the recommended events to the group. 4) Optionally if the group member wants he/she can view the interest rates of the other group members and how many of the group members are available during the recommended event’s time.
Entry condition	The group member needs to be signed in. The group member needs to be the member of the group that he/she is viewing the recommendations of.
Exit condition	The group member closes the recommended events tab.

Use case name	Buy Tickets for the Created Events
Participating Actor	Group Member, Payment Services Provider

Flow of Events	1) Group member opens the page of the event that is created by the event runner. 2) Group member clicks the buy tickets button. 3) Group member enters the users who he/she is buying a ticket for. 4) Group member enters the credit card information and other necessary information doing a purchase online. 5) Group member clicks the confirmation button. 6) Payment services provider performs the money transaction. 7) Event runner receives the money paid by the group member.
Entry condition	Group member needs to be signed in.
Exit condition	Group member receives the confirmation of the tickets bought for the event. Confirmation mail sent to every user group member's mail has bought a ticket for.

Use case name	Cancel Created Events
Participating Actor	Event Runner, Payment Service Provider
Flow of Events	1) Event runner views the event created by them. 2) Event runner enters the event detail page. 3) Event runner clicks on cancel button. 4) Event runner confirms cancellation by accepting confirmation popup. 5) Event is removed from the Perfent. 6) System notifies the users about event cancellation. 7) Payment service provider ensures that ticket buyers get their money back.

Entry condition	Event runner is signed in and the creator of the event.
Exit condition	Event runner confirms cancellation and is redirected to the event view page.

Use case name	View Created Events
Participating Actor	Event Runner
Flow of Events	1) Even runner opens his/her profile page. 2) Event runner clicks the created events tab. 3) Event runner views the created events and information about the event. (Basic event information + bought tickets etc.)
Entry condition	Event runner needs to be signed in. Event runner needs to be the creator of the events he/she is viewing.
Exit condition	Event runner closes the created events tab.

Use case name	Create Event
Participating Actor	Event Runner
Flow of Events	1) Event runner clicks on create tab to switch to create page. 2) Event runner fills in information related to the event (name, price, etc.). 3) Event runner selects an optional

	group to propose the event to. 4) Event runner clicks on save button to save the event or clicks on cancel button to discard.
Entry condition	Event runner needs to be signed in.
Exit condition	Event runner saves the event to is redirected to the events page.

3.5.1.3 Schedule Operations

Use case name	Indicate Between Which Dates They Are Not Available
Participating Actor	User
Flow of Events	1) User clicks on Schedule page 2) User clicks on Indicate Unavailable Dates 3) A pop-up is shown to the user 4) User selects start date and end date 5) User will not get notify for the events during the interval
Entry condition	User needs to be signed in User must have an imported schedule
Exit condition	User successfully selects the interval and closes the pop-up

Use case name	Hide Schedule Information from Other Users
Participating Actor	User

Flow of Events	1) User clicks on Schedule page 2) User clicks on a cell in the schedule 3) A pop-up menu for cell settings is displayed to the user 4) User selects hide information 5) Information related to the cell is no more displayed to other users
Entry condition	User needs to be signed in User must have an imported schedule
Exit condition	The information of the selected cell is hidden from other users

Use case name	Mark Occupied or Available Slots in the Schedule
Participating Actor	User
Flow of Events	1) User clicks on Schedule page 2) User clicks on a cell in the schedule 3) A pop-up menu for cell settings is displayed to the user 4) User selects Mark as Occupied or Available 5) The slot is displayed as Occupied or Available
Entry condition	User needs to be signed in User must have an imported schedule Selected slot shouldn’t overlap with the user’s attending events

Exit condition	The slot is shown as occupied or available in the User's schedule

Use case name	View Slots that Attended Events Occupy in the Schedule
Participating Actor	User
Flow of Events	1) User clicks on Schedule page 2) User clicks on Attended Events 3) The schedule with the user's previously attended events is shown to the user
Entry condition	User must be signed in User must have an imported schedule
Exit condition	The previously attended events are shown

Use case name	View the Time Slots the Proposed and Recommended Events Occupy in the Group Schedule
Participating Actor	Group Member
Flow of Events	1) Group Member clicks on Schedule page 2) Group Member clicks on Group Schedule

	3) Group Member selects the relevant group 4) Group Member clicks on Proposed and Recommended Events 5) The group's combined schedule with proposed and recommended events is shown to the user
Entry condition	Group member needs to be signed in Group member must have a group with a valid combined schedule
Exit condition	The schedule with proposed and recommended events is shown to the user

Use case name	View Combined Group Schedule
Participating Actor	Group Member
Flow of Events	1) Group Member clicks on Schedule page 2) Group Member clicks on Group Schedule 3) Group Member selects the relevant group 4) The group's combined schedule is shown to the user
Entry condition	Group member needs to be signed in Group member must have a group with a valid combined schedule
Exit condition	The combined group schedule is shown to the user

Use case name	Combine Imported Schedules
Participating Actor	User
Flow of Events	1) User clicks on Schedule page 2) User clicks on combine with another schedule 3) Schedule import pop-up is shown to the user 4) User imports a new schedule 5) Schedules are combined and shown in the page
Entry condition	User must be signed in User must have a schedule in the system
Exit condition	The combined schedule is shown to the user

Use case name	Synchronize Schedule with Third Party Applications
Participating Actor	User, Schedule Services Provider
Flow of Events	1) User Clicks on Schedule page 2) User clicks on Synchronize button 3) The User's Schedule is updated according to the imported schedule with the services of schedule services provider.
Entry condition	User needs to be signed in
Exit condition	The schedule is successfully updated OR

	Imported calendar is no longer available

Use case name	Import Schedule From Third Party Applications
Participating Actor	User
Flow of Events	1) User clicks on Schedule page 2) User clicks Import Schedule button 3) A pop-up is displayed to the user 4) User selects a site from the displayed options in which they have their schedule is stored 5) User's schedule is imported from the website of their choice.
Entry condition	User needs to be signed in
Exit condition	User successfully imports the calendar and is redirected to the schedule page.

3.5.1.4 General Operations

Use case name	Request to Create an Event Runner Account
Participating Actor	Event Runner
Flow of Events	1) Event runner clicks the create event runner account button. 2) Event runner enters the necessary information to create an event runner account. (This might include id, organized past events and evidence that shows past events are organized by the person) 3) Event runner clicks the submit button to submit the application.
Entry condition	Event runner needs to open the Perfent site.

Exit condition	The event runner receives the confirmation of their application sent to Perfent systems.

Use case name	Verify Event Runner Accounts
Participating Actor	Perfent Support
Flow of Events	1) Perfent support clicks the tab that they can see event runner account creation requests. 2) Perfent support clicks a request to see its details. 3) Perfent support evaluates the information post at event runner account request. 4) Perfent support verifies the event runner account request.
Entry condition	Perfent support needs to be signed in.
Exit condition	Perfent support receives the event runner account verified prompt.

Use case name	Block Notifications From the Application
Participating Actor	User
Flow of Events	1) User opens the profile page. 2) User clicks the “turn off notifications” button.
Entry condition	User needs to be signed in.
Exit condition	User receives the notifications turned off prompt.

Use case name	User Edits Profile
Participating Actor	User
Flow of Events	1) User clicks “Edit Profile” button 2) A pop-up for editing is displayed to the user 3) User changes the information 4) The necessary process for changing the particular information is performed by the site.
Entry condition	User is logged in. User opens Profile page.
Exit condition	User receives the “your profile is successfully edited” prompt.

Use case name	User Registration
Participating Actor	User
Flow of Events	1) User enters required information 2) User chooses a password 3) User clicks register button
Entry condition	User opens the registration page.

	User doesn’t have another account logged in.
Exit condition	Event runner receives the confirmation of their application sent to Perfent systems.

Use case name	User Login
Participating Actor	User
Flow of Events	1) User enters a valid username or email and password. 2) User is navigated to the home page.
Entry condition	User is not already logged in User opens login page
Exit condition	User successfully logs in OR User navigates to another page OR User closes registration page

Use case name	View Profile
Participating Actor	User

Flow of Events	1) User opens profile page 2) The information stored for the profile is displayed to the user
Entry condition	User is logged in User opens Profile page
Exit condition	User navigates to another page OR User logs out OR User closes the page

3.5.1.5 User Matching Operations

Use case name	Receive User Recommendations For Joining Events Together
Participating Actor	User
Flow of Events	1) User clicks on My Events page 2) User clicks on Upcoming 3) User clicks on Recommended Users 4) Recommended Users for that event are displayed to the user 5) User either accepts or ignores the recommendations
Entry condition	User must be signed in User must have a certain upcoming event that they are going
Exit condition	Users group is updated according to their choice

Use case name	Allow/Block Yourself from User Recommendations
Participating Actor	User
Flow of Events	1) User clicks on Profile page 2) User clicks on Preferences 3) User selects one of the options, Allow or Block, from the User Recommendations Setting
Entry condition	User must be signed in
Exit condition	User's profile preferences are successfully updated

Use case name	View Previously Matched Users
Participating Actor	User
Flow of Events	1) User clicks on Profile page 2) User clicks on History 3) User clicks on Previously Matched Users 4) Users that were matched with the user are displayed to the user
Entry condition	User must be signed in
Exit condition	Previously matched users are displayed to the user

Use case name	Evaluate Previously Matched Users
Participating Actor	User

Flow of Events	1) User clicks on Profile page 2) User clicks on History 3) User clicks on Previously Matched Users 4) Users that were matched with the user are displayed to the user 5) User clicks Evaluate button near one of the users 6) A pop-up is displayed in the browser 7) User fills the form in the pop-up 8) User clicks Submit button
Entry condition	User must be signed in User must have previously matched users
Exit condition	The evaluation form is submitted to the database

Use case name	Report Matched Users
Participating Actor	Event Attendee
Flow of Events	1) Event attendee clicks on Profile page 2) Event attendee clicks on History 3) Event attendee clicks on Previously Matched Users 4) Users that were matched with the Event attendee are displayed to the user 5) Event attendee clicks Report button near one of the users 6) A pop-up is displayed in the browser 7) Event attendee fills the form in the pop-up 8) Event attendee clicks Submit button
Entry condition	Event Attendee must be signed in

	Event Attendee must have previously matched users
Exit condition	The report for is submitted to the database

Use case name	View Reports
Participating Actor	Perfent Support
Flow of Events	1) Perfent support clicks on Reports page 2) The reports are displayed to the support
Entry condition	Perfent support must be signed in There must be reports in the system
Exit condition	The reports are shown to the Perfent Support

Use case name	Evaluate Reports
Participating Actor	Perfent Support
Flow of Events	1) Perfent support clicks on Reports page 2) The reports are displayed to the support 3) Perfent support clicks on Evaluate button near one of the reports 4) A pop-up is displayed to the Perfent Support 5) Perfent support fills the form 6) Perfent support clicks on Submit button
Entry condition	Perfent support must be signed in There must be reports in the system

Exit condition	Action is taken by the system depending on the result of the report

3.5.1.6 Event Attending Operations

Use case name	Create a List of Items to Bring
Participating Actor	Event Attendee
Flow of Events	1) Event attendee clicks on My Events 2) Event attendee clicks on Upcoming 3) Event attendee's upcoming events are shown 4) Event attendee clicks on Create List of Items 5) A pop-up for creating the list is displayed 6) Event attendee fills the form in the pop-up 7) Event attendee clicks Submit
Entry condition	Event attendee must be signed in Event attendee must have an upcoming event
Exit condition	The list is submitted for the event

Use case name	Assign Group Members to Bring Items
Participating Actor	Event Attendee

Flow of Events	1) Event attendee clicks on My Events 2) Event attendee clicks on Upcoming 3) Event attendee's upcoming events are shown 4) Event attendee clicks on Create List of Items 5) A pop-up for creating the list is displayed 6) Event attendee assigns other group members to bring the items 7) Event attendee clicks Submit
Entry condition	Event attendee must be signed in Event attendee must have an upcoming event Event attendee must have a group
Exit condition	Group members get assigned to bring items

Use case name	Receive Notifications About Bringing Items
Participating Actor	Event Attendee
Flow of Events	1) Event attendee clicks on My Events 2) Event attendee clicks on Upcoming 3) Event attendee's upcoming events are shown 4) Event attendee clicks on Settings button near an upcoming event 5) Event attendee toggles the setting for receiving notifications about bringing items 6) Event attendee clicks Submit
Entry condition	Event attendee must be signed in Event attendee must have an upcoming event

Exit condition	Event attendee starts or stops getting notifications for bringing items

Use case name	Post Videos and Photos from the Event to the Group's that Event's Feed
Participating Actor	Event Attendee
Flow of Events	1) Event attendee clicks on My Events 2) Event attendee clicks on All 3) All events of Event Attendee are shown 4) Event attendee clicks on Feed button near an event 5) The feed of the event is shown 6) Event attendee clicks post 7) Event attendee uploads a video or a photo of their choice 8) Event attendee clicks Submit
Entry condition	Event attendee must be signed in Event attendee must have an event on the list
Exit condition	The photo or the video that was uploaded is shown on the feed of that event

Use case name	View the Least Time Consuming Ways to Reach Event Location
Participating Actor	Event Attendee

Flow of Events	1) Event attendee clicks on My Events 2) Event attendee clicks on Upcoming 3) Upcoming events of Event attendee are shown 4) Event attendee clicks on Directions button near an event 5) A map with directions to event location are shown to the user
Entry condition	Event Attendee must be signed in Event Attendee must have an upcoming event
Exit condition	The fastest is calculated and shown to the user

Use case name	Mark Events as Attended
Participating Actor	Event Attendee
Flow of Events	1) Event attendee clicks on Events 2) The list of Events in the system are shown to the Event Attendee 3) Event attendee clicks Mark as Attended button near an event
Entry condition	Event Attendee must be signed in
Exit condition	The marked event is displayed in Event Attendee's My Events page

Use case name	Anonymously Post Photos and Videos from the Event in the Page of the Event
Participating Actor	Event Attendee

Flow of Events	1) Event attendee clicks on My Events 2) Event attendee clicks on All 3) All events of Event Attendee are shown 4) Event attendee clicks on Photos button near an event 5) All photos from the event are displayed to the event attendee 6) Event attendee clicks Post Anonymous Picture or Video 7) Event attendee uploads a picture or a video 8) Event attendee clicks on Submit button
Entry condition	Event attendee must be signed in Event attendee must have an event
Exit condition	The uploaded file is displayed to other users in the event page

Use case name	Rate Events
Participating Actor	Event Attendee
Flow of Events	1) Event attendee enters the events view tab. 2) Event attendee filters for past events or searches by event name. 3) User finds the event to rate. 4) User clicks on the event and enters the event detail page. 5) User clicks on the stars icon in this page and gives the rating.
Entry condition	User needs to be signed in and marked the event as attended.

	The user needs to ensure the system that he/she attended to event by marking it as attended
Exit condition	User rating is taken by the system and event rating is recomputed.

Use case name	Comment Events
Participating Actor	Event Attendee
Flow of Events	1) Event attendee enters the events view tab. 2) Event attendee filters for past events or searches by event name. 3) Event attendee finds the event to write a comment to. 4) Event attendee clicks on the event and enters the event detail page. 5) Event attendee clicks on the comment button. 6) A text box opens to write the views. 7) Event Attendee writes their comments and clicks publish button.
Entry condition	Event attendee needs to be signed in and marked the event as attended.
Exit condition	Event attendee comment is taken by the system.

3.5.2 Use-Case Model

Figure 1: Use case diagram of the Perfent system

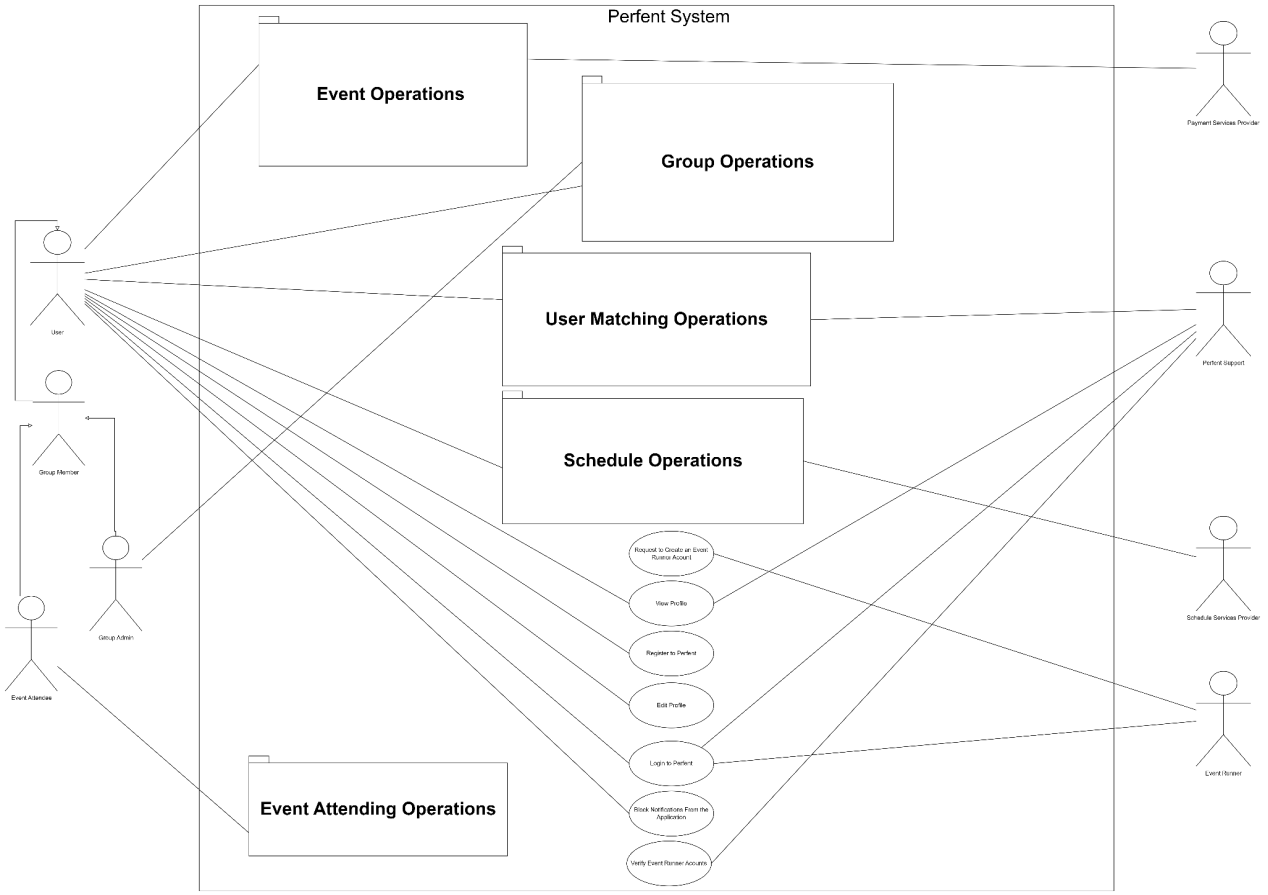


Figure 2: Use case diagram of user matching operations package

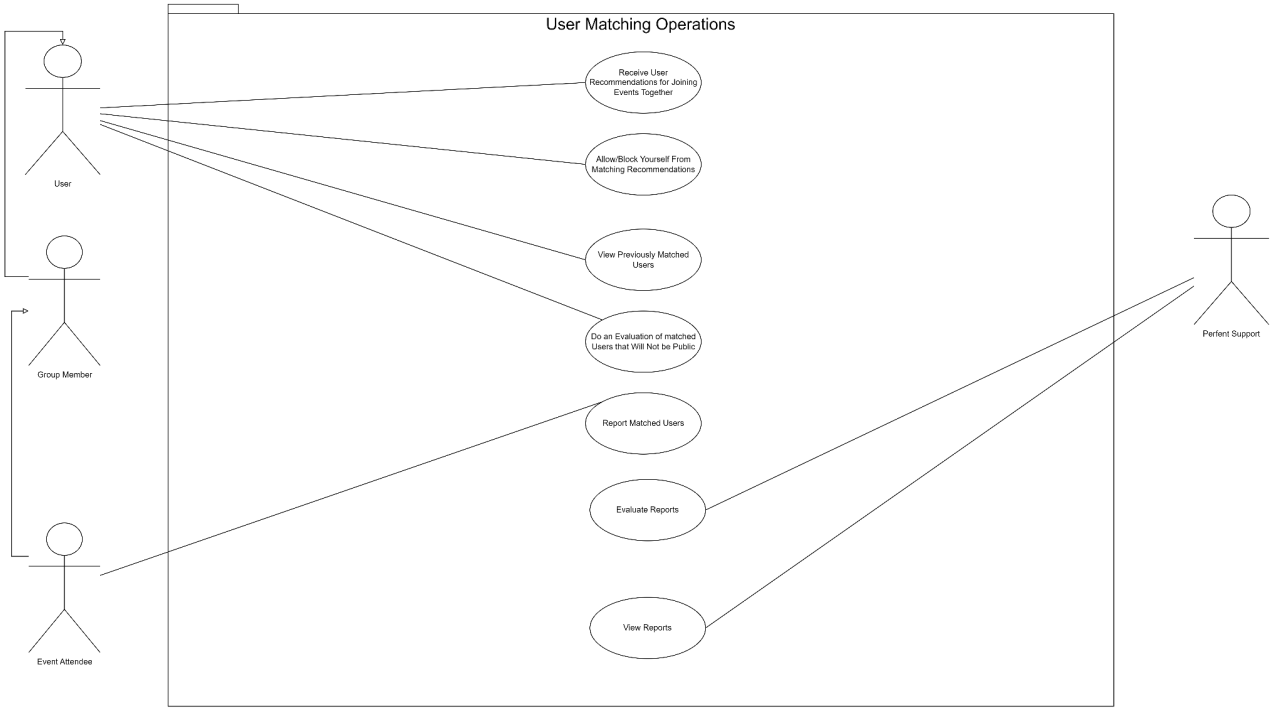


Figure 3: Use case diagram of schedule operations package

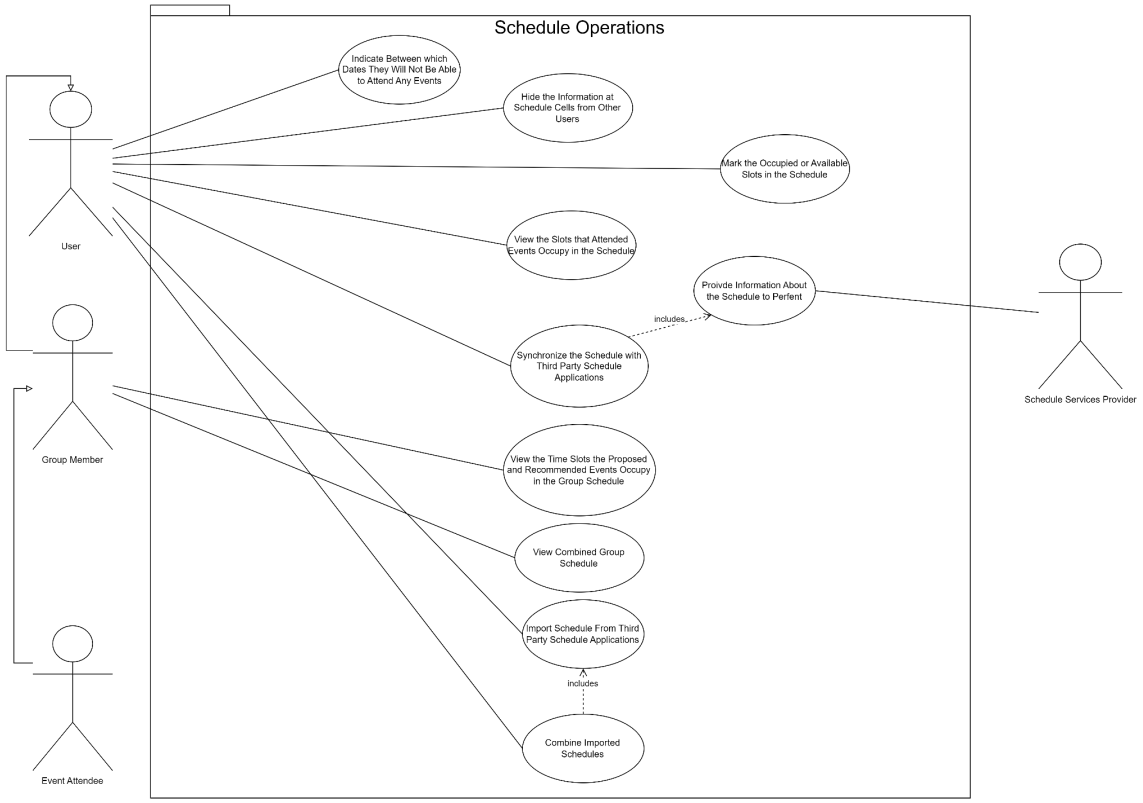


Figure 4: Use case diagram of event attending operations package

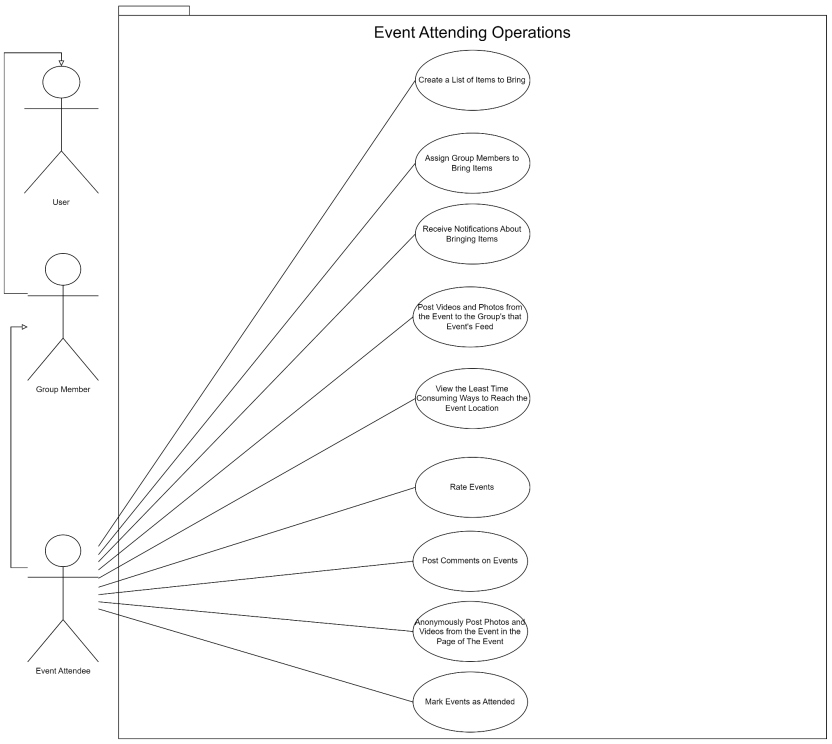


Figure 5: Use case diagram of group operations package

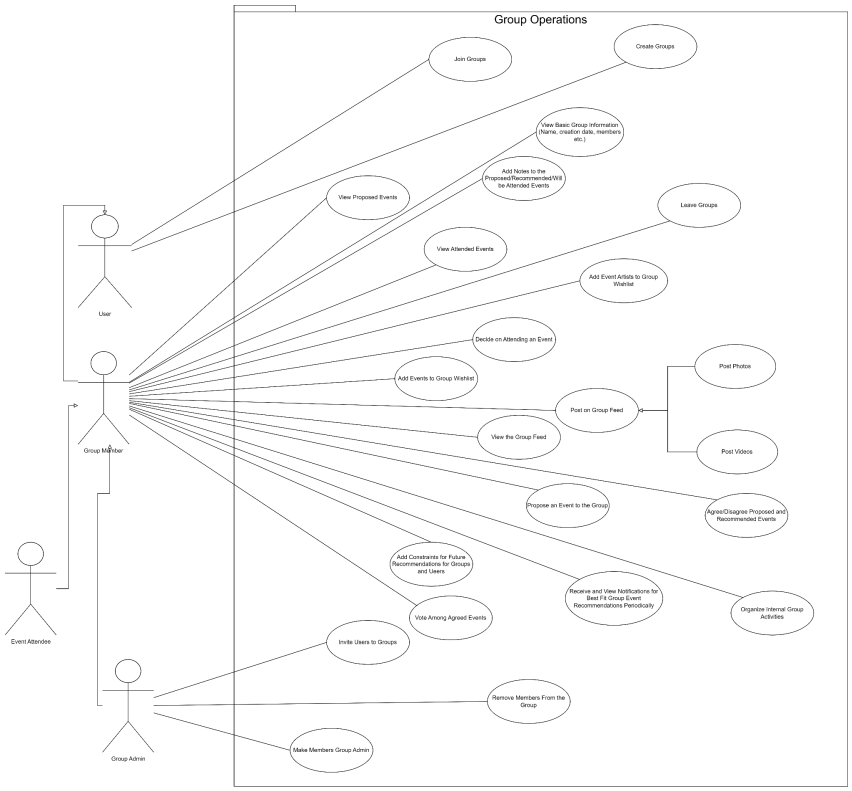
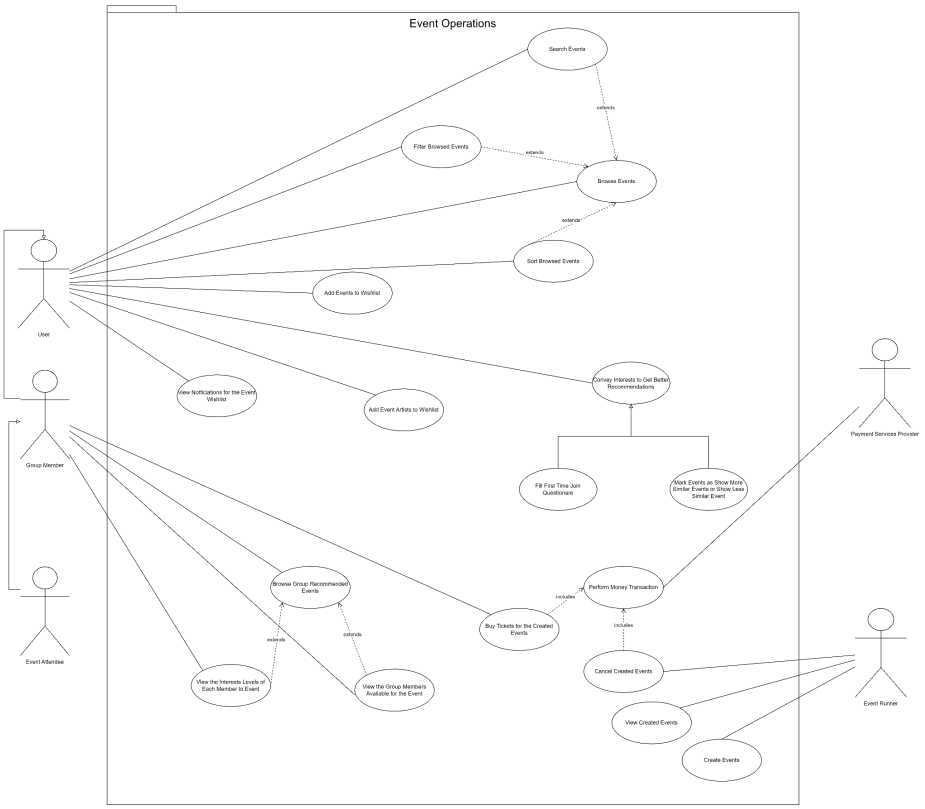
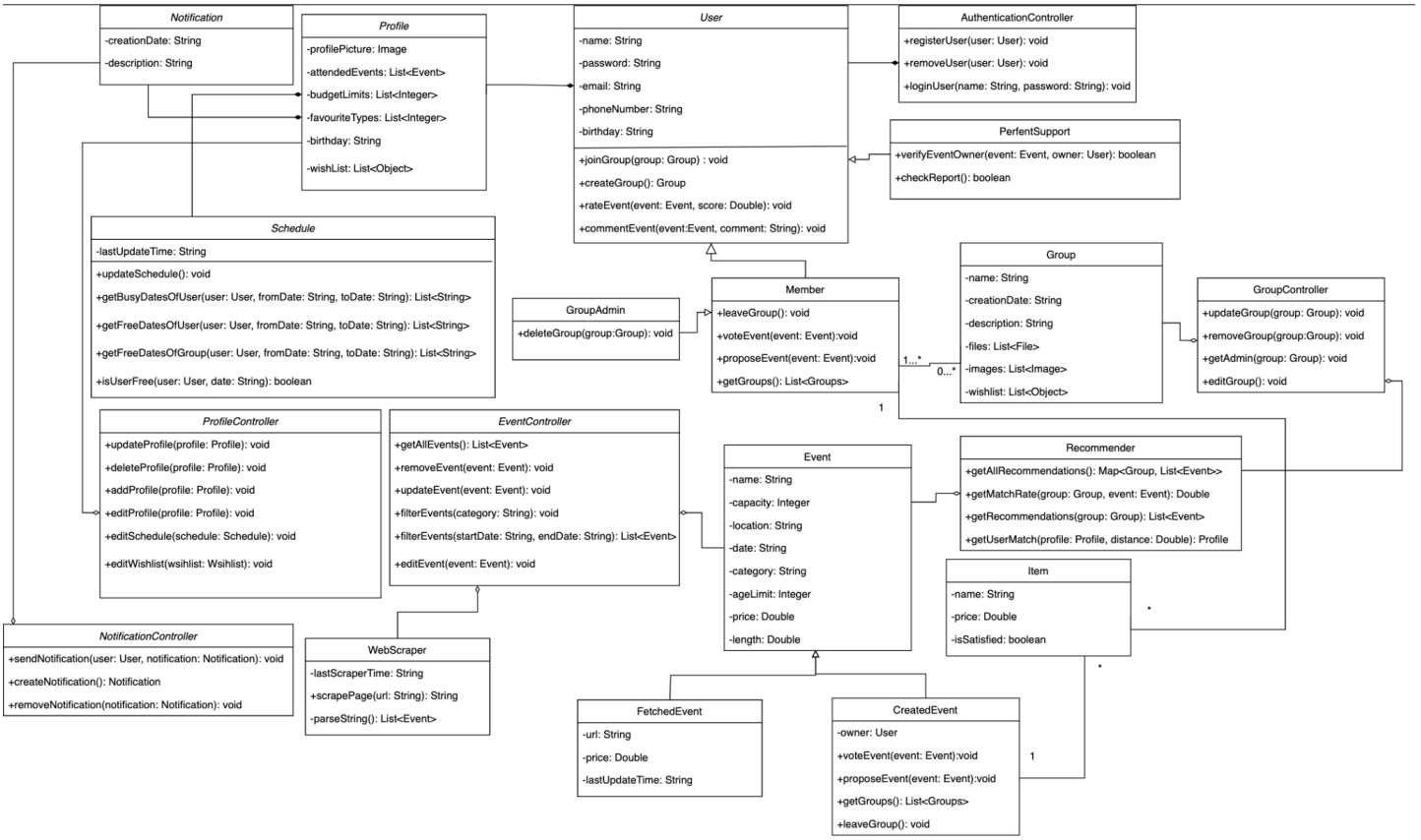


Figure 6: Use case diagram of event operations package



3.5.3 Object and Class Model

Figure 7: The Object and Class Model of Perfent



The class explanations are as follows:

- **User**

  This class models the registered users of the system.

  It has the password, email and name information to authenticate the user.

  Every user has a profile in the system.

- **Member**

  When users join a group, they become the members of that group. A group can have many members and a member can be part of many groups.

- **GroupAdmin**

  If a user creates a group, it becomes the admin of the group. Every admin is also a member of the group.

- **PerfentSupport**

PerfentSupport checks the reports of the users and verifies the event owners.

- **AuthenticationController**

This controller class registers and removes the users from the system.

This class also checks if the user can enter to the system in other words if the user information is correct.

- **Profile**

Every user in the system has a profile where they show their activities and preferences in the system.

Every user profile has a schedule and a wishlist.

Notifications appear in the profiles.

- **Schedule**

Every user uses a schedule to keep track of the events and to be able to get recommendations for their groups.

Every schedule is in the user profiles.

- **Profile Controller**

This controller class manages the profile actions.

It helps to update, edit profiles, calendars and wishlist

- **Group**

Groups are the target class of the application whom the recommendations are created for.

Every group consists of many members.

Every group has related documents for their events.

- **GroupController:**

This controller class manages to control the groups.

It updates, removes and edits groups.

It consists of a recommender to get the recommendations for the groups.

- **Event**

  Events are the activities that users can browse and attend together.

  Events have different attributes such as name, location and category.

  Every event can be attended by many users and many users can attend different events.

  Events can be fetched from the web from existing activities or users can create their own events as well.

- **CreatedEvent**

  These are the events created by individual users.

- **Item**

  These are the materials required by the createdevents.

  Every createdEvent can have many items needed and every item gets assigned to a member.

- **FetchedEvent**

  These are the events that are created by organizations and fetched from the web.

  These events have price information and the link url to go to the actual website.

- **Recommender**

  This class uses machine learning to create recommendations to the groups.

  If provides the recommendations to the group controller.

- **EventController**

  This class controls the event related operations such as updating and removing an event.

- **WebScraper**

  This class fetches the events from the web and provides them to the application.

- **Notification**

  This class represents the information provided to the users regularly to inform them on the new suggestions at the system.

- **NotificationController**

This class controls the notifications that are sended to the users regarding the events and other suggestions in the system.

3.5.4 Dynamic Models

3.5.4.1 Activity Diagrams

Figure 8: Activity Diagram showing how a group decides on an event to attend

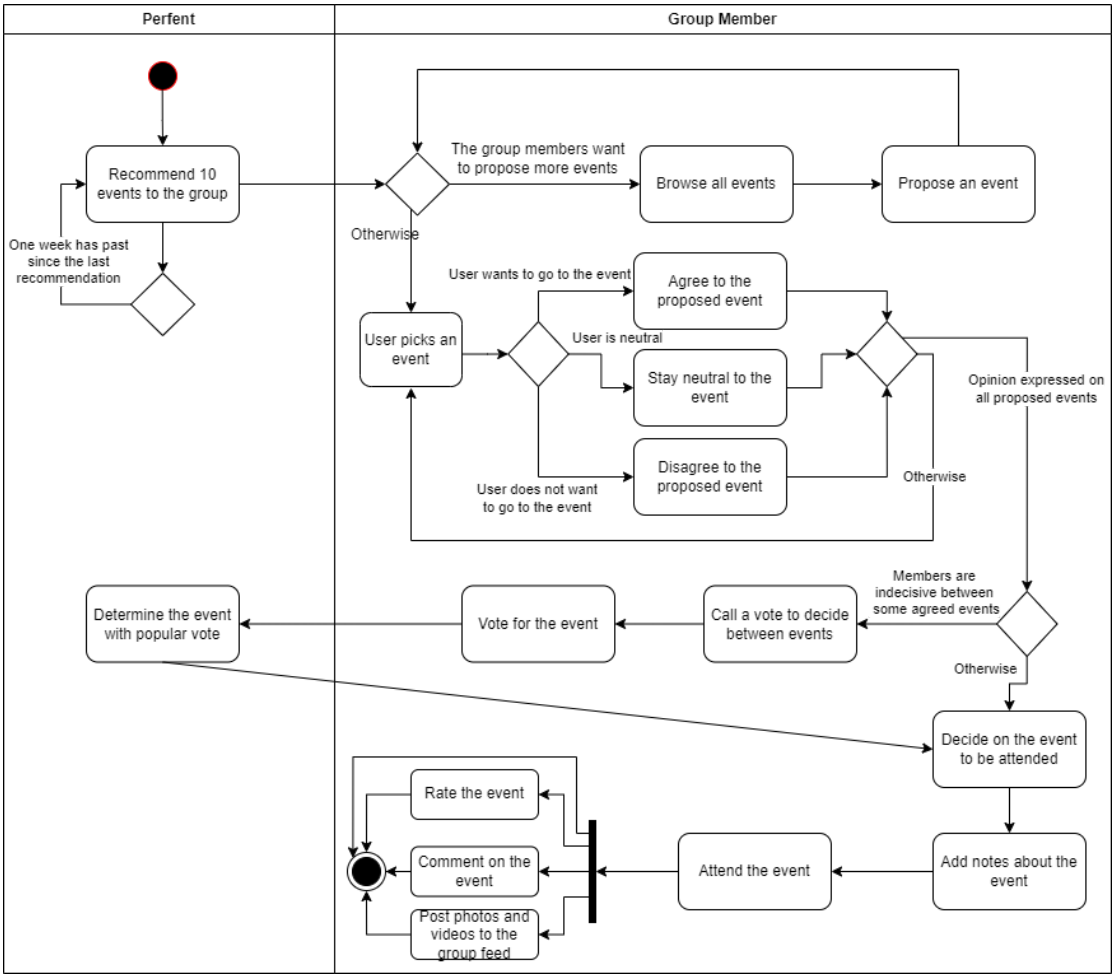


Figure 9: Activity Diagram showing how the Web Scraper gathers event data

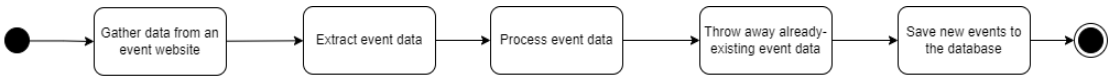


Figure 10: Activity diagram of recommendation system process of recommending

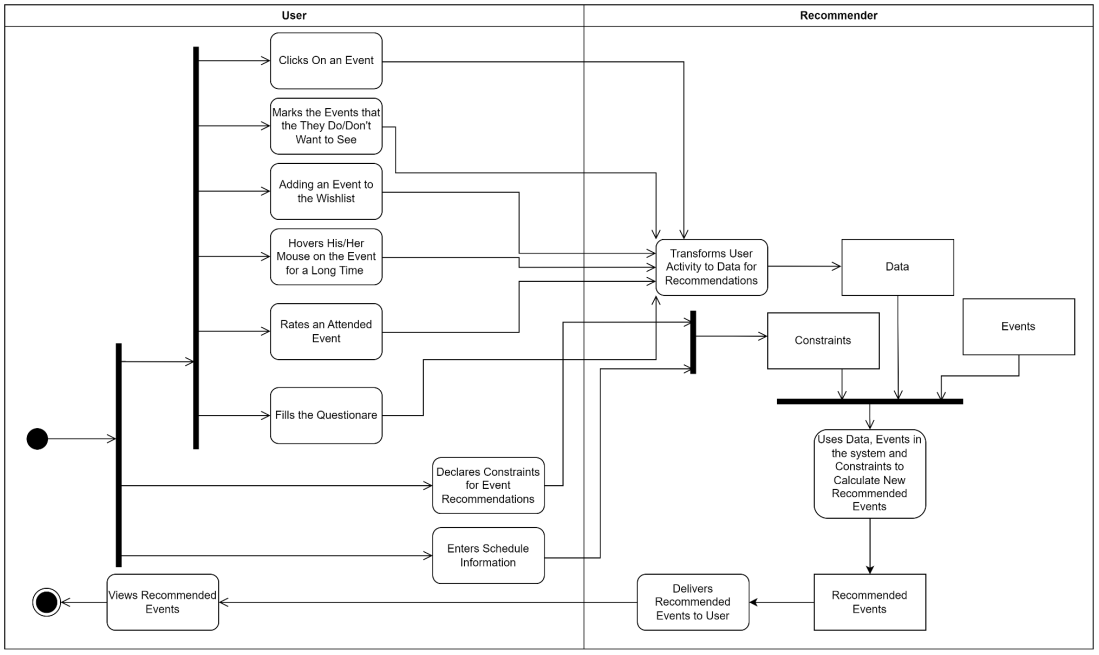
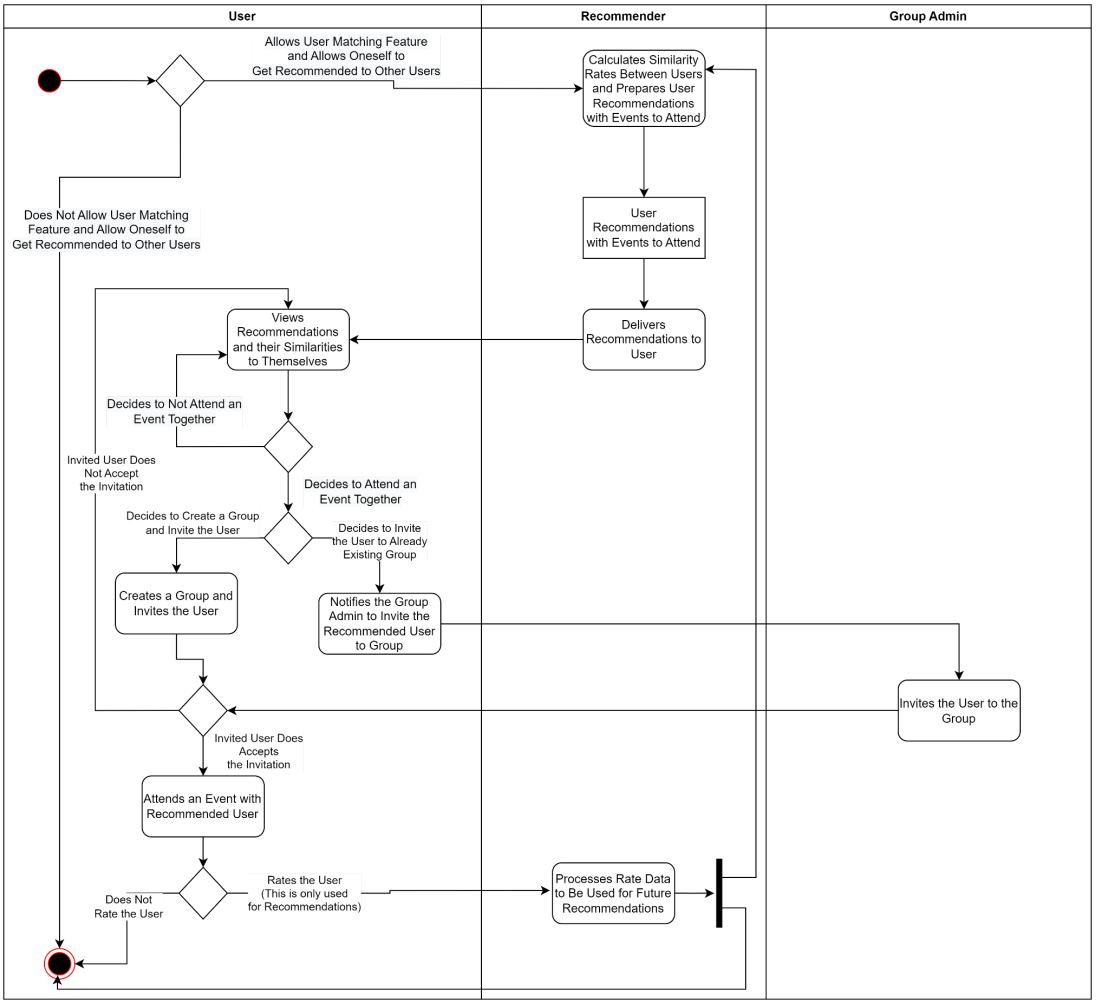
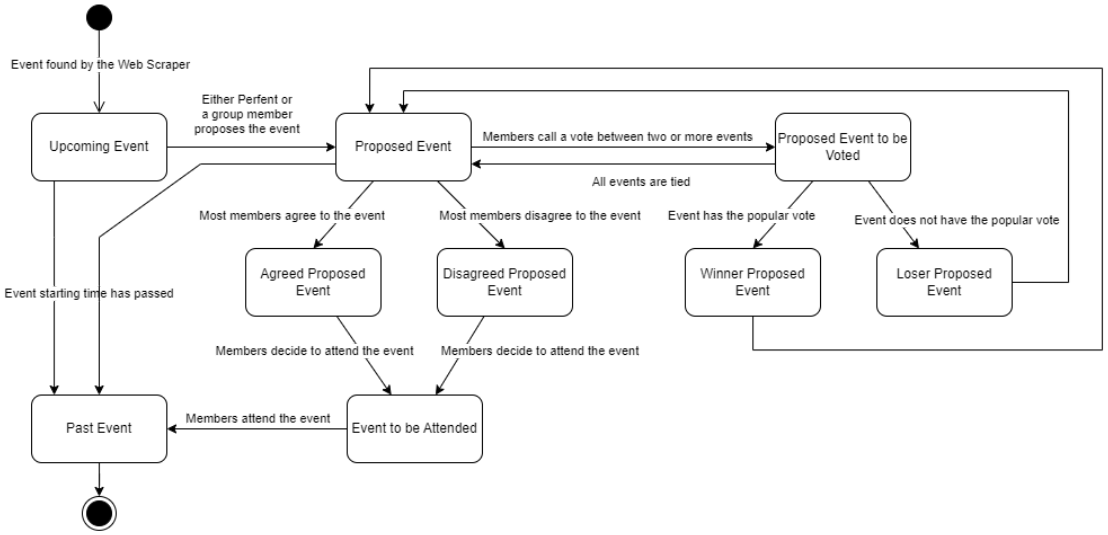


Figure 11: Activity diagram of user matching process



3.5.4.2 State Diagrams

Figure 12: Event State Diagram



The state diagram shows the states of an event relative to a particular group. The diagram shown above is slightly simplified to be easy to understand: In the actual product, there is a transition from each state to the “Past Event” state if the event starting time passes. Another simplified part is the following: The diagram shows that an event can be voted only if it is in the “Proposed Event” state. However, the same transitions between “Proposed Event” and “Proposed Event to be Voted” should also be between “Agreed Proposed Event” and “Proposed Event to be Voted” as well as “Disagreed Proposed Event” and “Proposed Event to be Voted.” The transitions are not shown but described here as not to complicate the diagram. Another point is that there can be transitions between “Agreed Proposed Event” and “Disagreed Proposed Event” depending on the agreed/disagree situation.

3.5.4.3 Sequence Diagrams

Figure 13: Sequence diagram of adding a user to a group

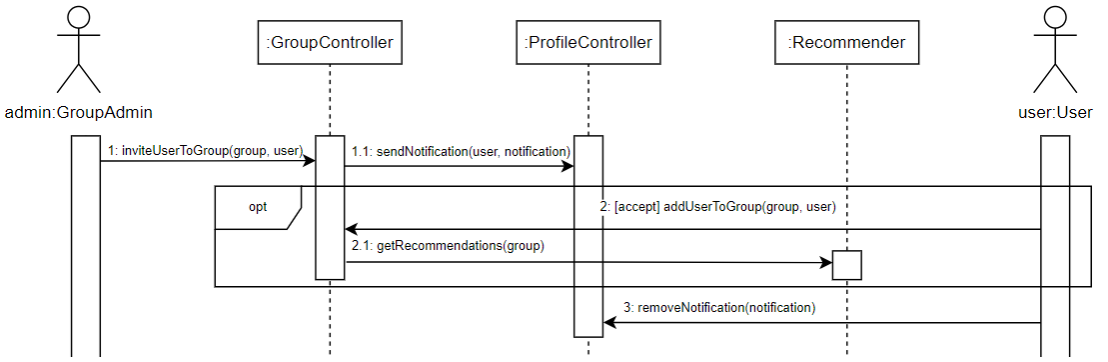
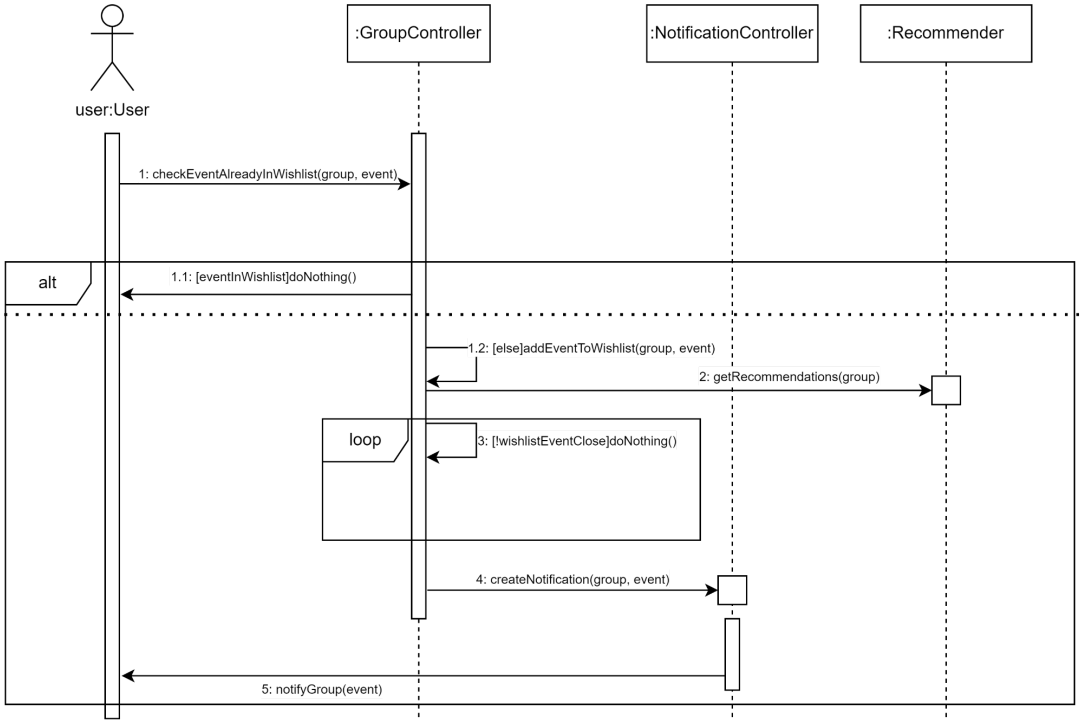


Figure 14: Sequence diagram of adding an event to the group wishlist



3.5.5 User Interface

To see User Interfaces use the link.

Figure 15: Perfent Splash Screen

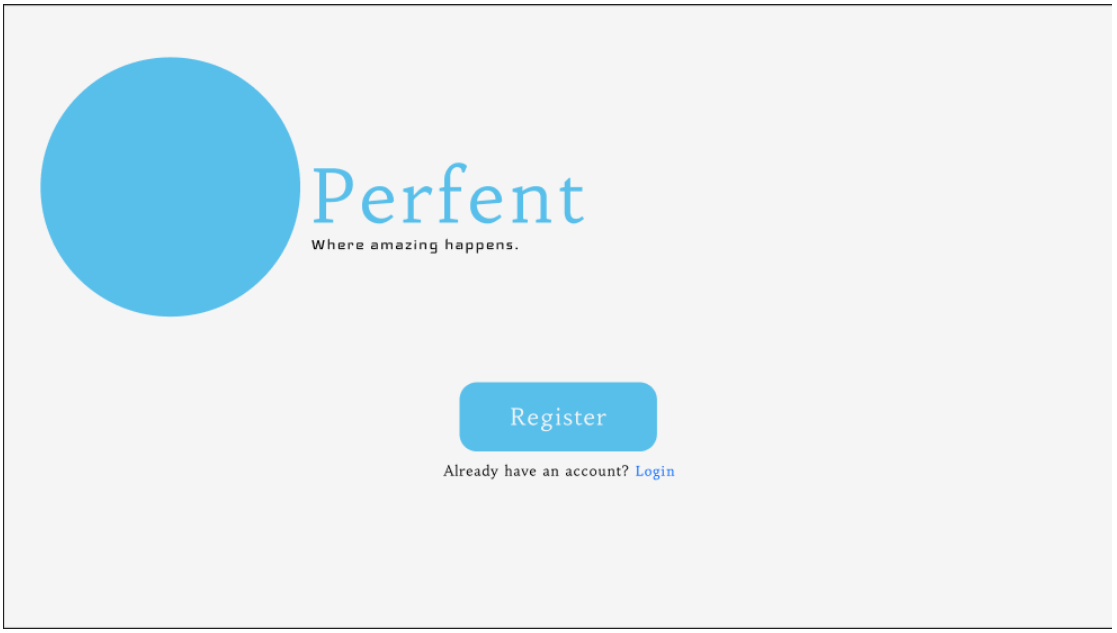


Figure 16: Perfent Login Screen

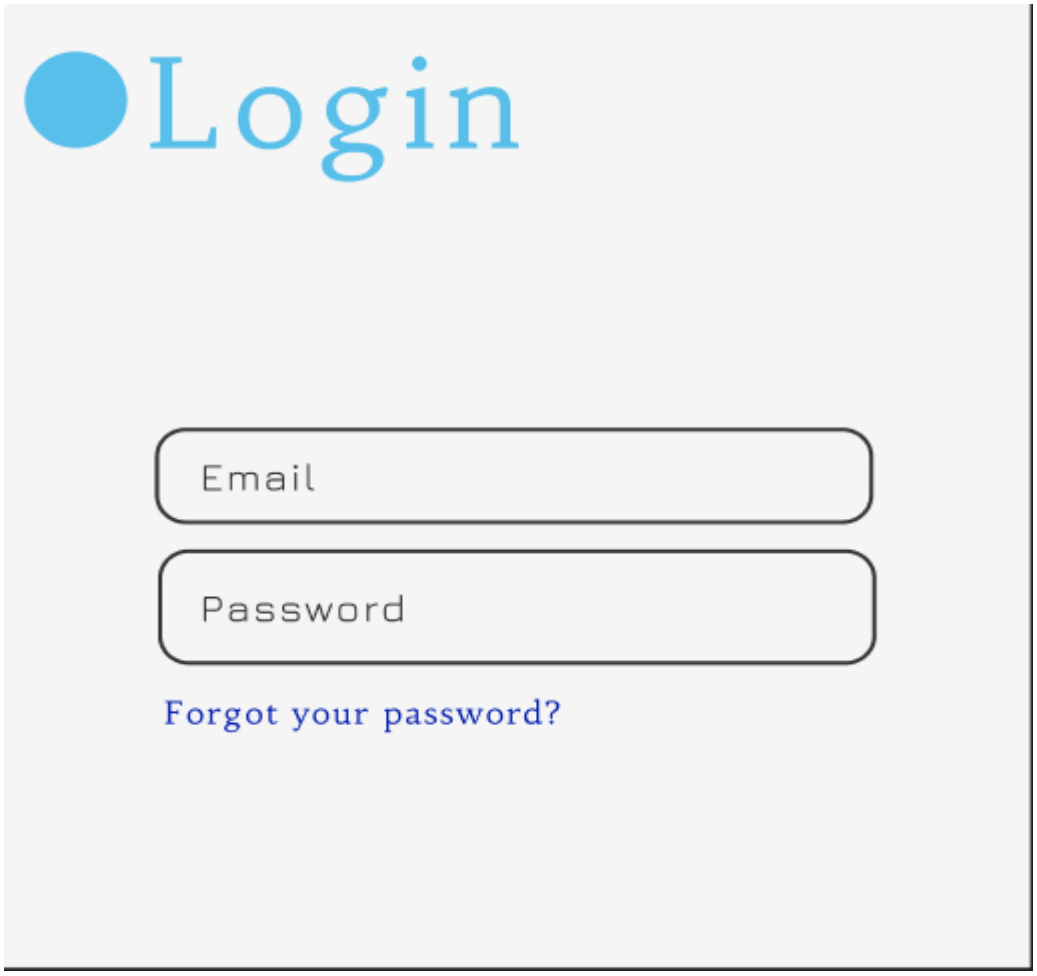


Figure 17: Perfent Register Screen

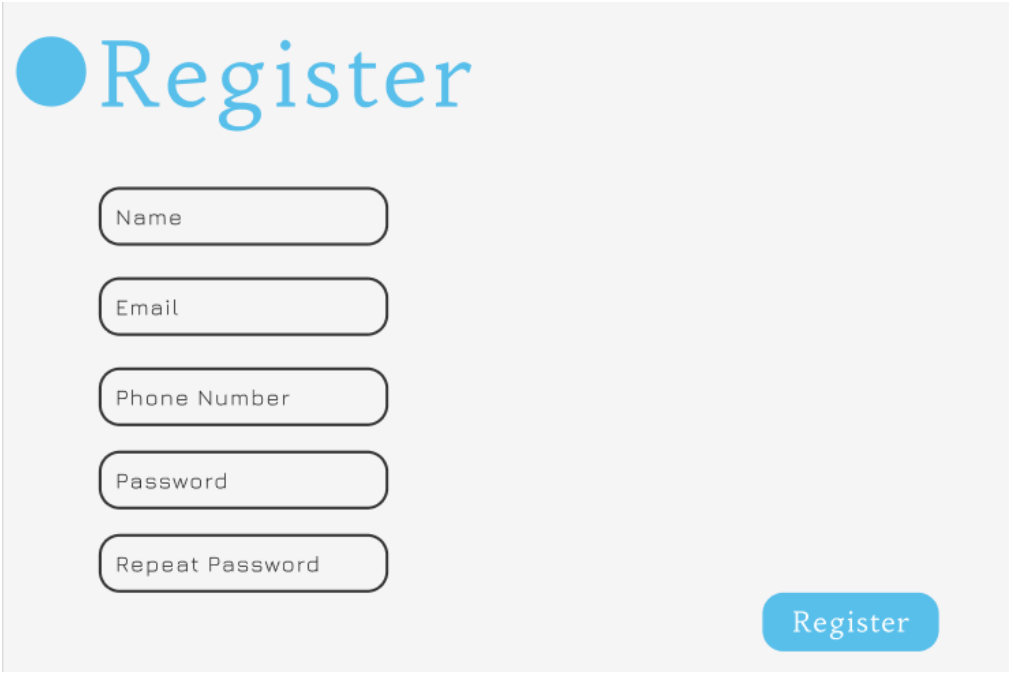


Figure 18: Browse Events Page

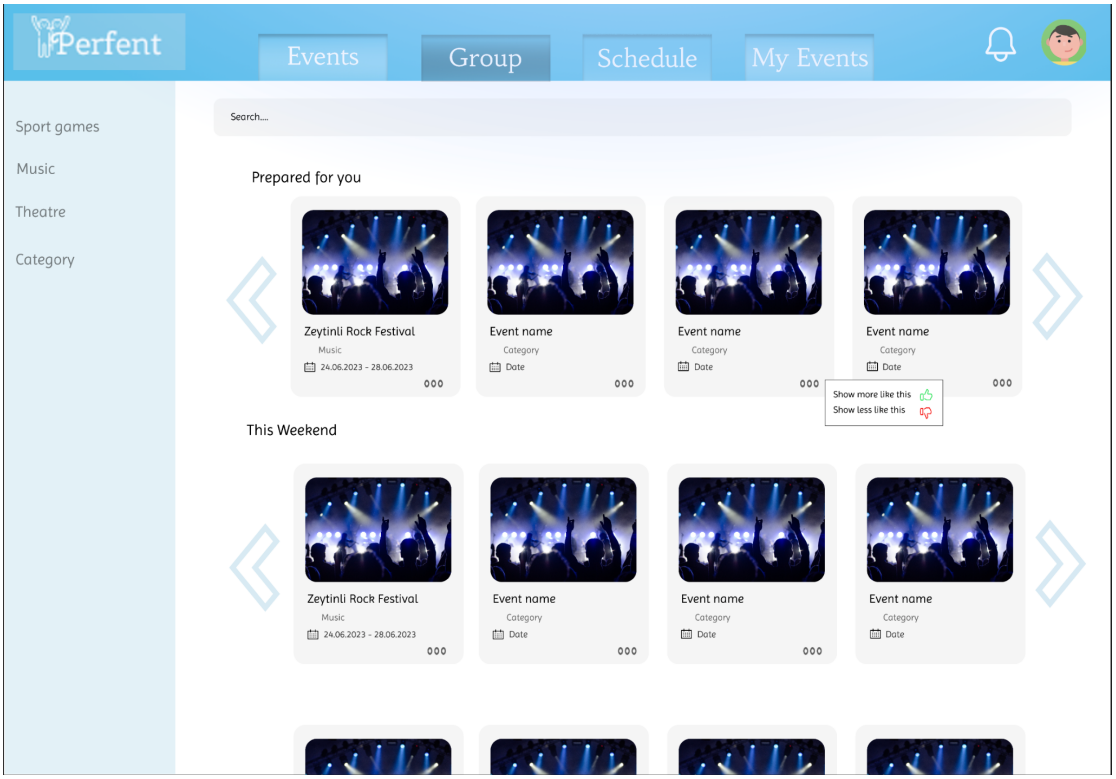


Figure 19: My Events Page

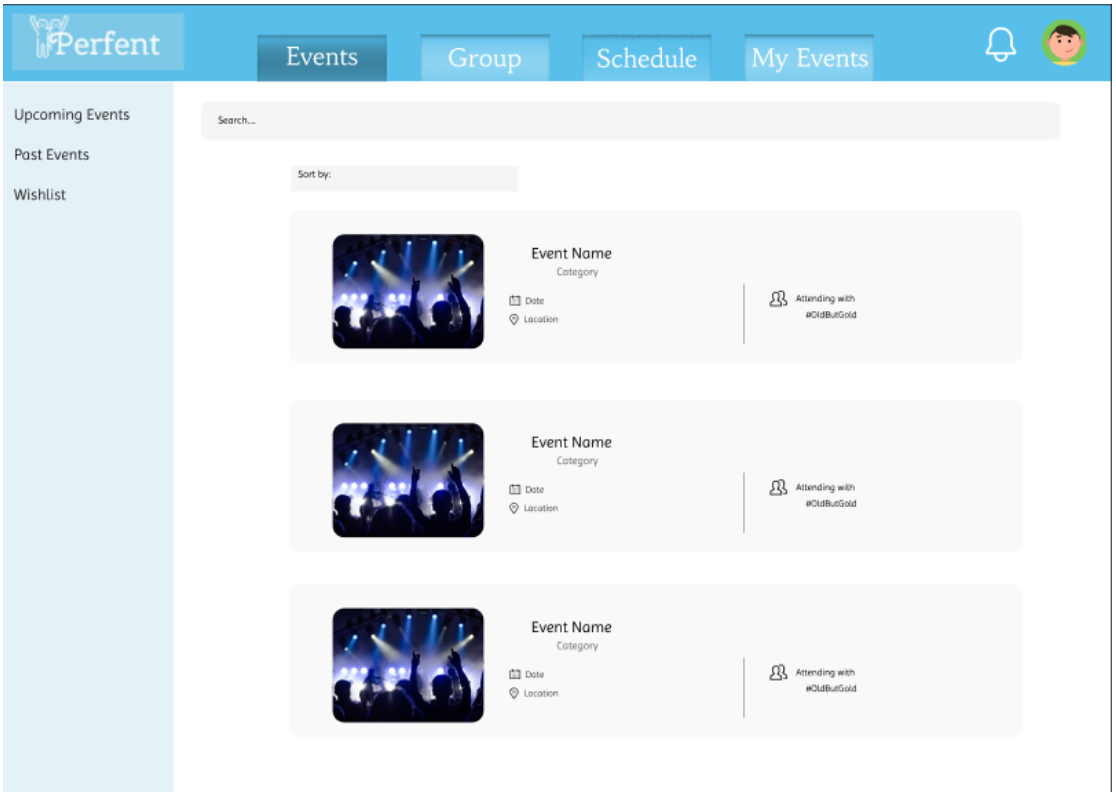


Figure 20: Group Event View

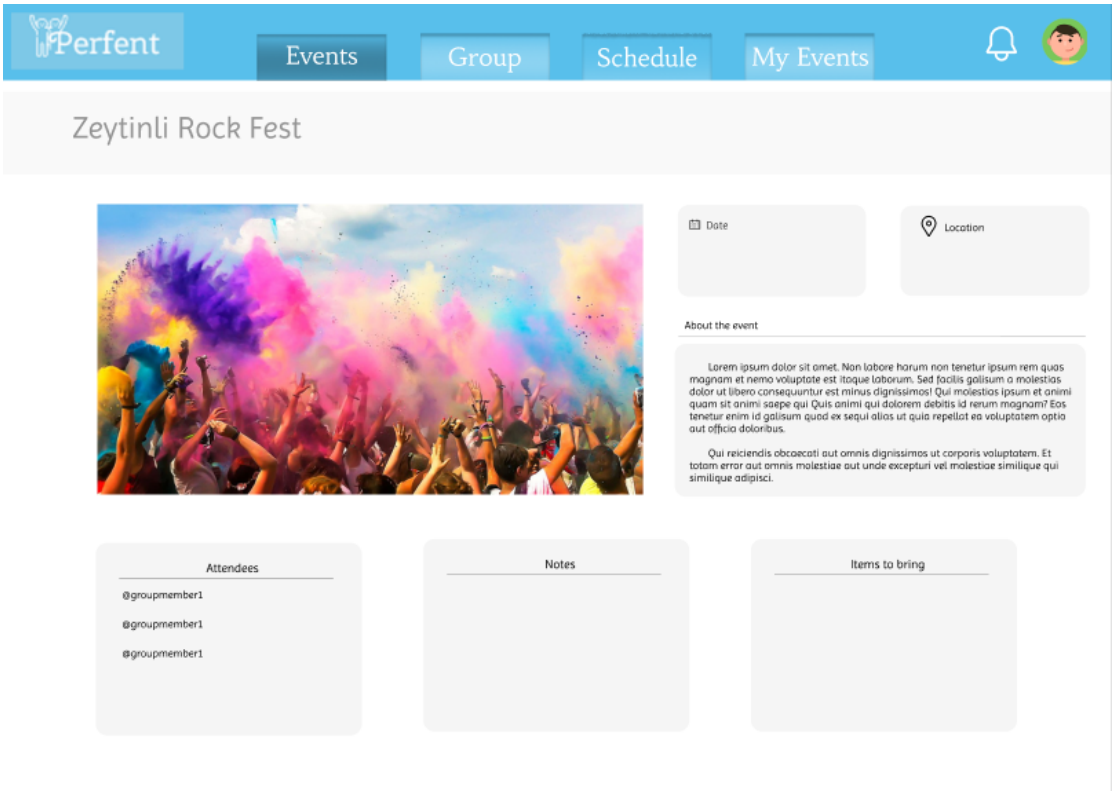


Figure 21: Upcoming Event View

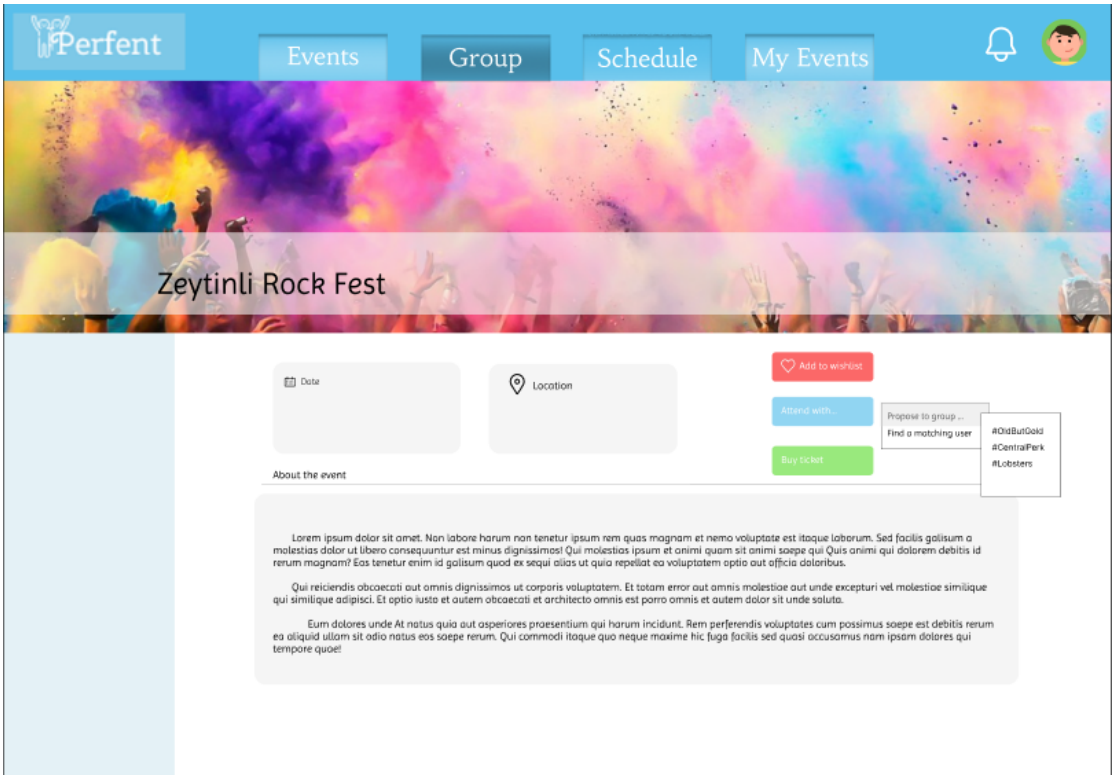


Figure 22: Past Event View

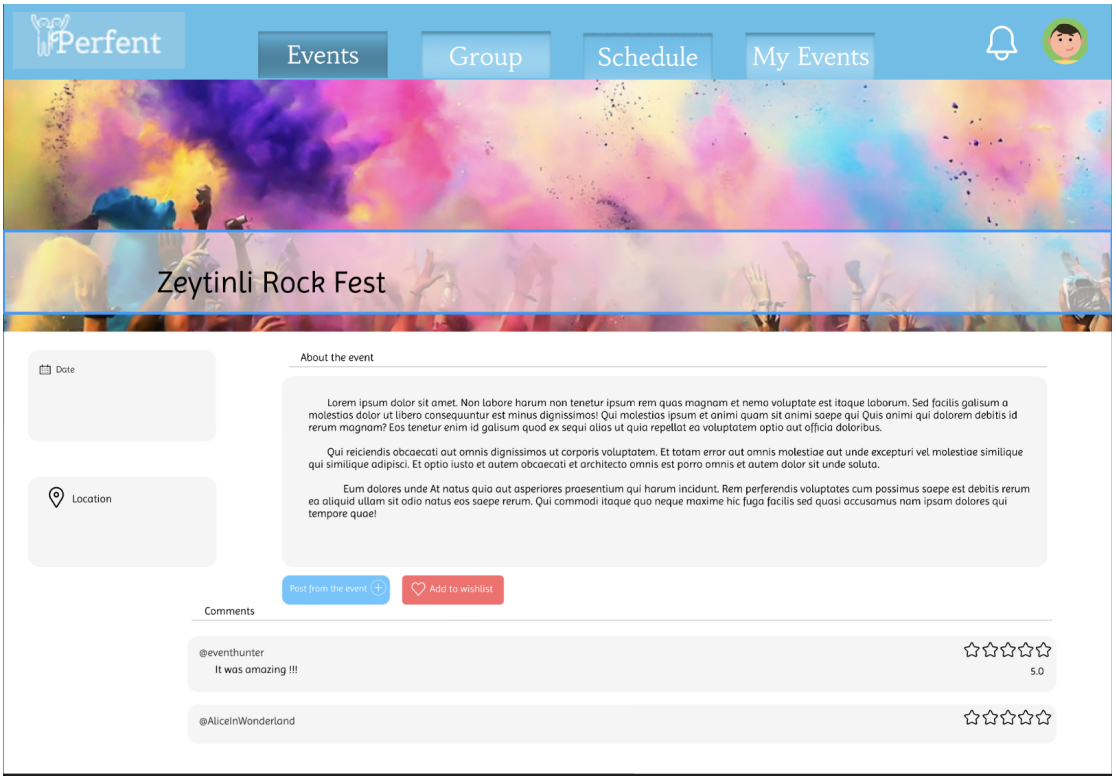


Figure 23: Schedule

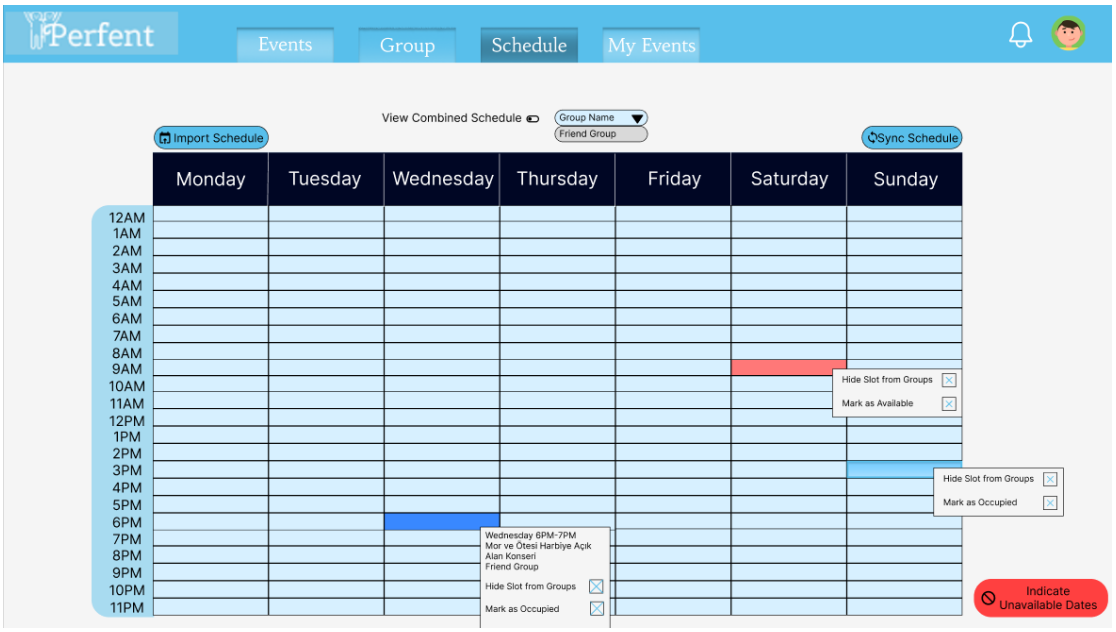


Figure 24: Indicate Unavailable Dates Pop-up

Figure 25: Group Recommended & Proposed Events

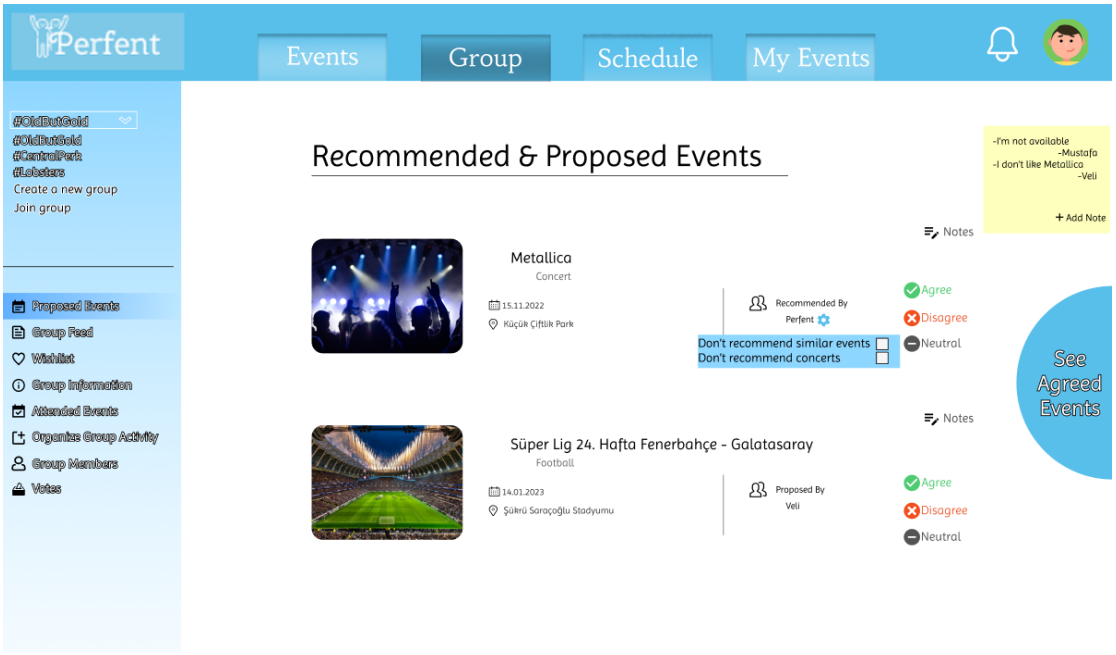


Figure 26: Group Agreed Events

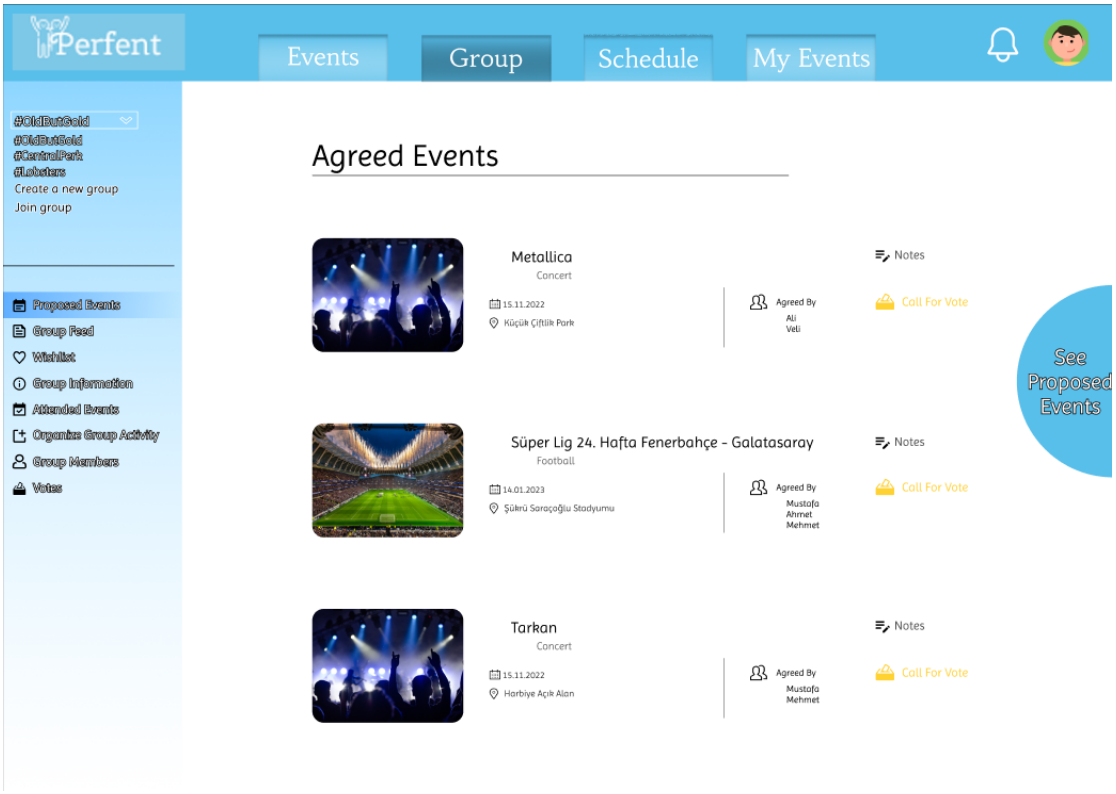


Figure 27: Vote Pop-up

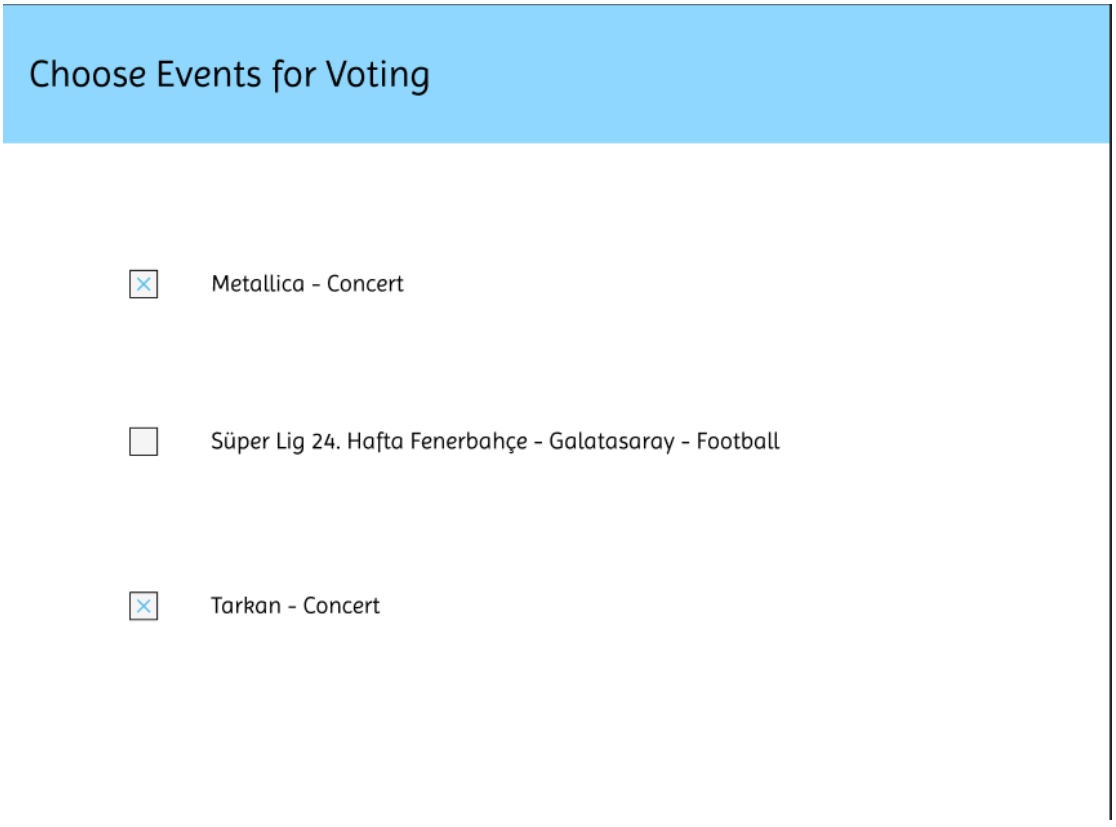


Figure 28: Ongoing Votes

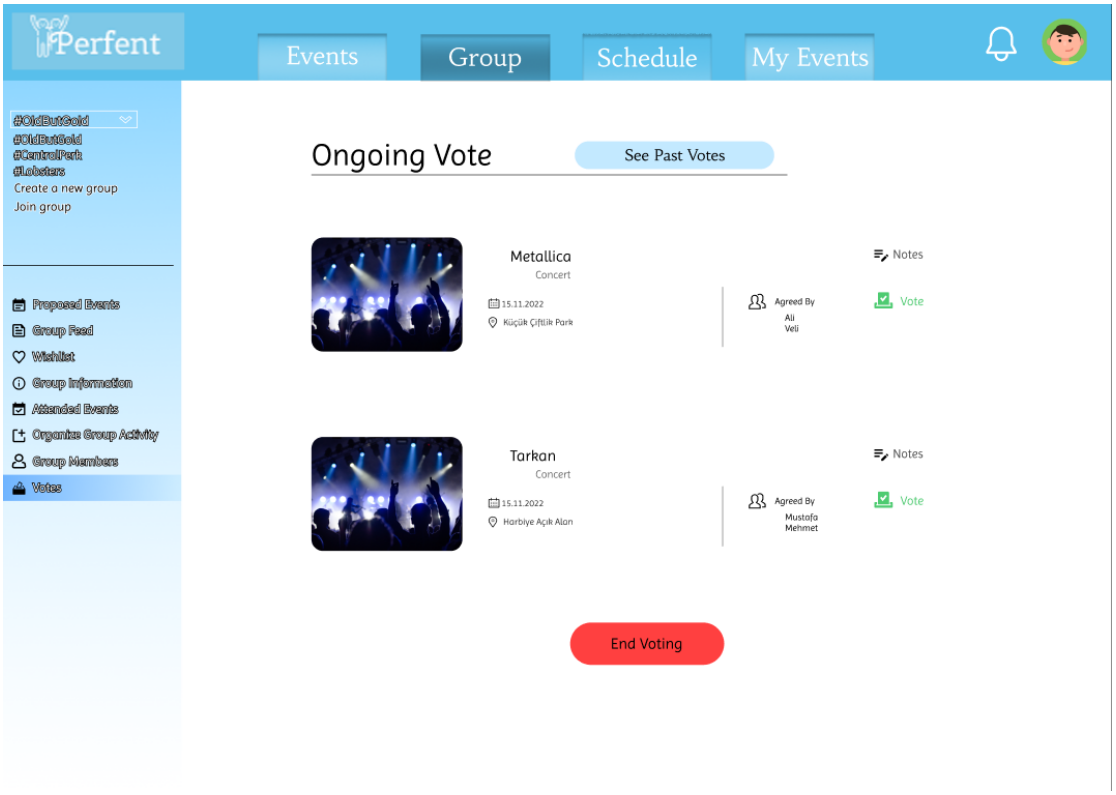


Figure 29: Past Votes

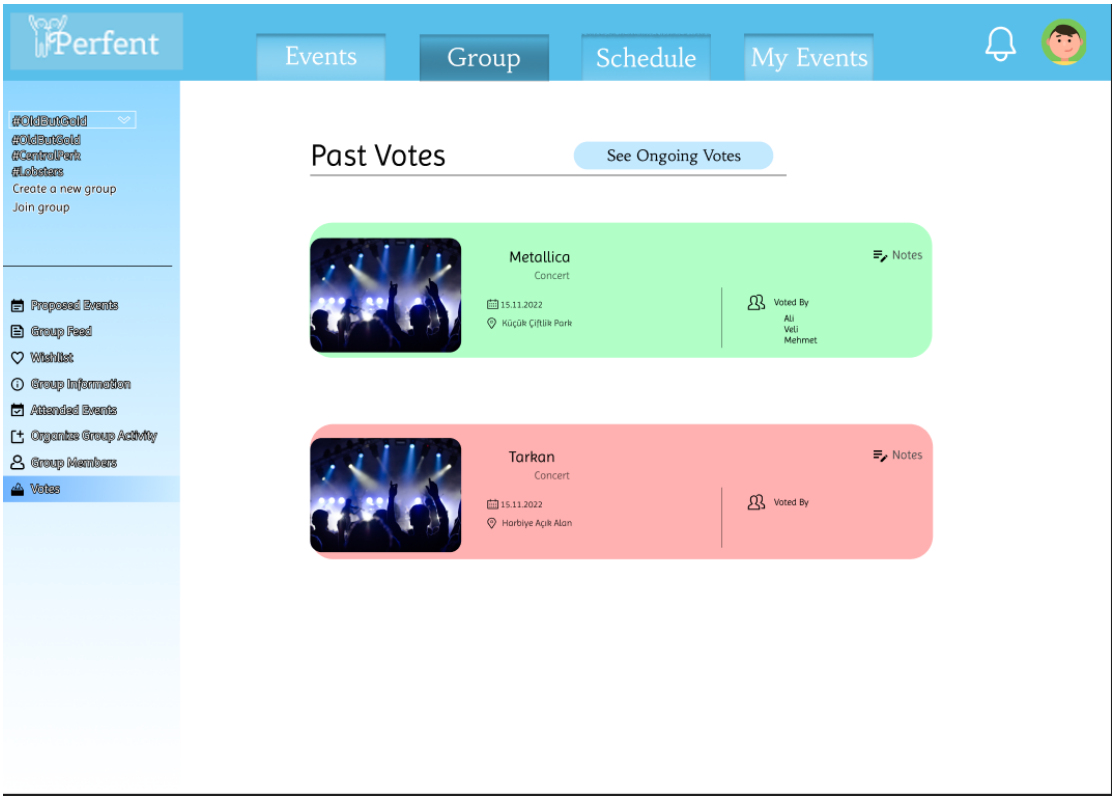


Figure 30: Group Information Pop-up

Figure 31: Group Feed

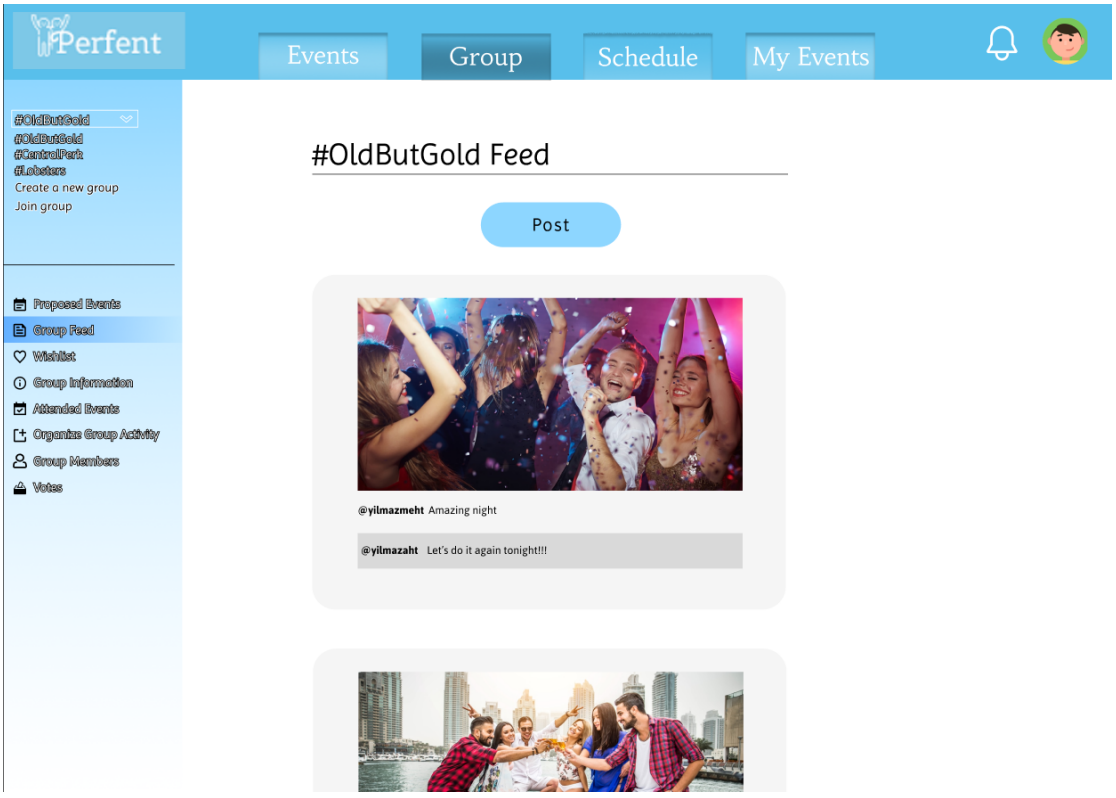


Figure 32: Group Wishlist

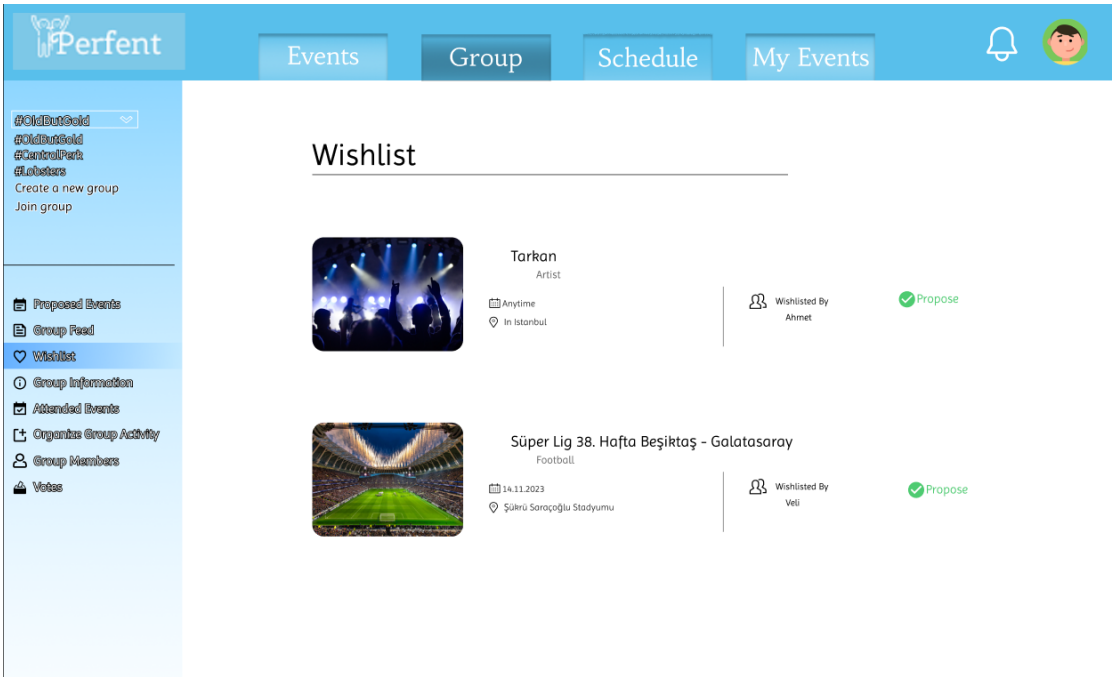


Figure 33: Group Members

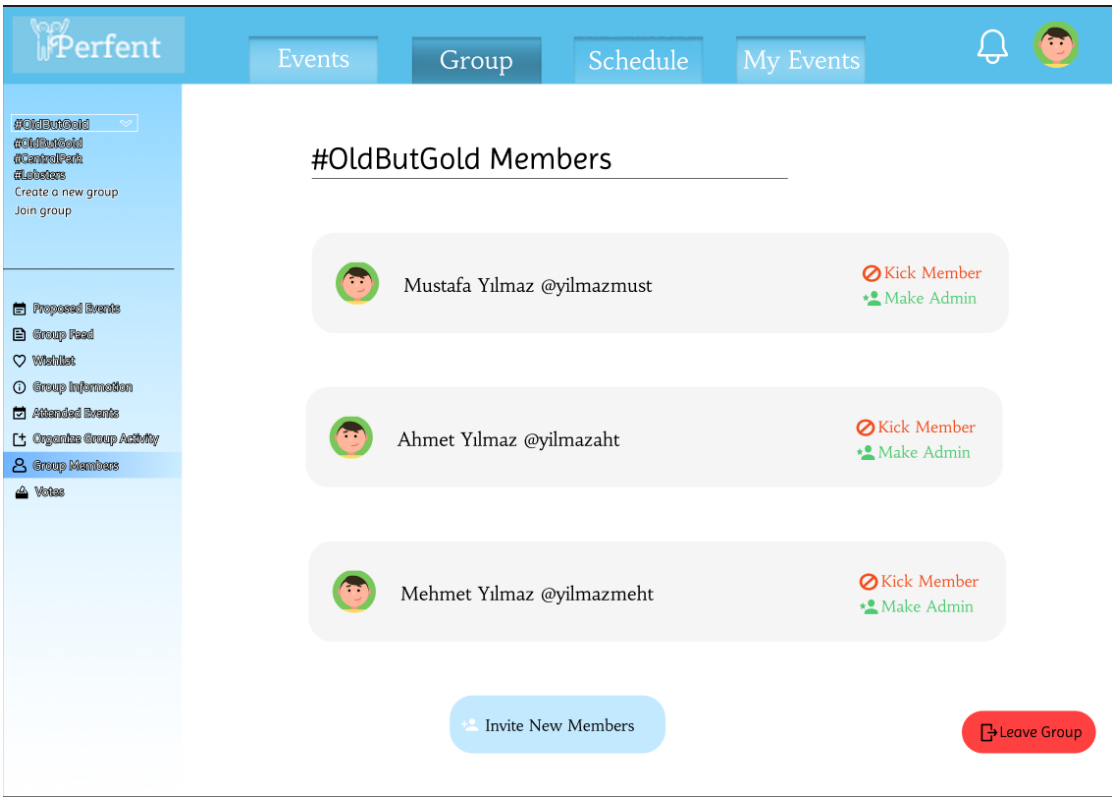


Figure 34: Profile page

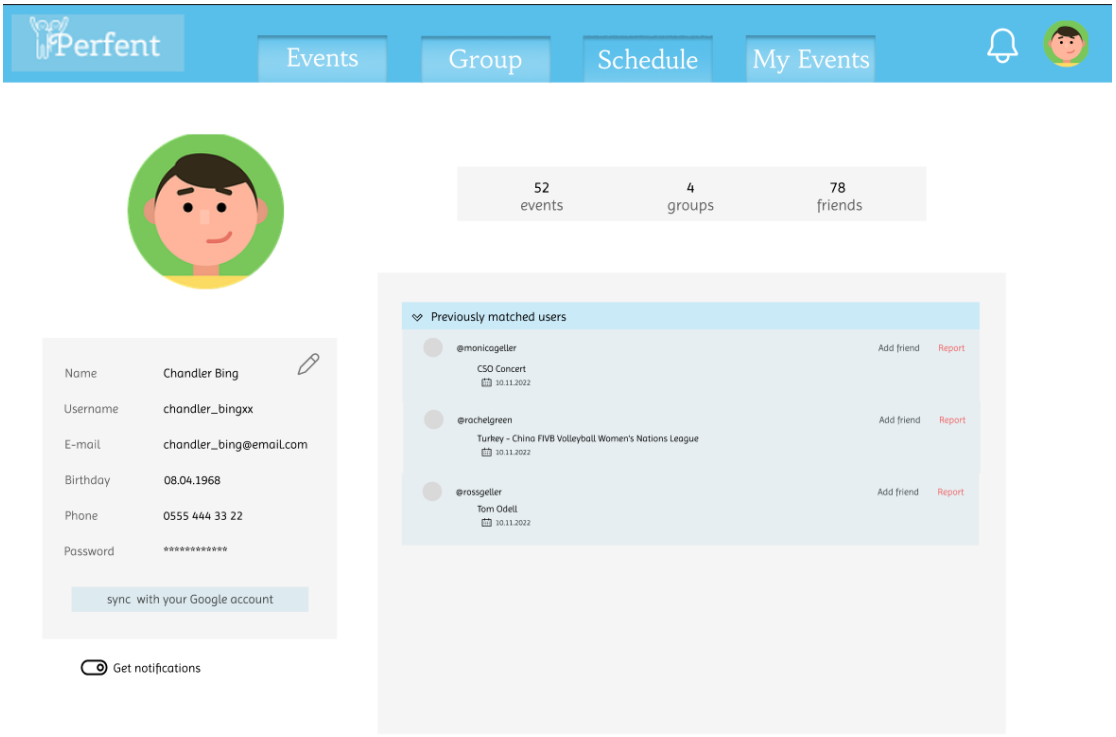
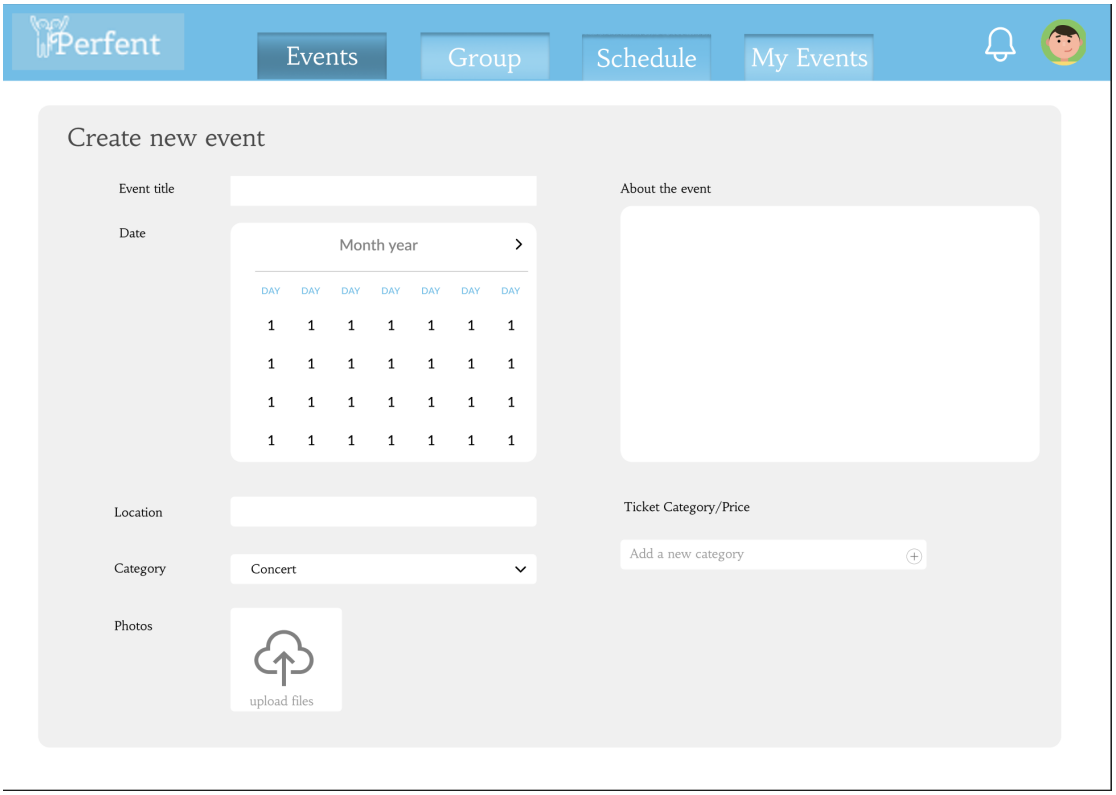


Figure 35: Create event page



4 Other Analysis Elements

4.1 Consideration of Various Factors in Engineering Design

In the engineering design of this project, the first consideration was the modularity of the application. Because of the fact that the application has a client-server architecture, the modularity between these components are important for detection of possible issues in terms of maintenance and portability. Furthermore, modularity is important inside these components as well. Especially at the server side, modularity is quite significant for project development because the server side has three different functionalities and to have an easy development process and to avoid problems in future, modular architecture will be helpful. Due to the fact that modularity is crucial for the application, the other main consideration was using technologies which will help us to easily assemble and disassemble the components. From this perspective, using Docker is strongly encouraged within the group members. In addition to that, using technologies which are easy to communicate with is important as well. To succeed this, Spring and React, which have been used by developers many times to implement server-client architecture, are decided to be used. The other consideration in the design was the compliance with the standards to reduce the risks for future problems. For example, the standard language for machine learning is python and because of that we

have decided to follow this common convention and use Python as the programming language for the recommendation system of the application. The last concern on the engineering design was related to high output and efficiency. To achieve this, we again benefited from modularity. Instead of performing data fetch and analysis steps directly in the web server, we decided to split this into further modules. This way,the overload on the server will be reduced and the client side will get the results in a fast and efficient manner.

4.2 Risks and Alternatives

As in all projects, there are some initial risks involved. In this section, these risks are identified, analyzed, and proper management strategies and alternatives are listed. Each risk is assigned a likelihood between one of “very low,” “low,” “moderate,” “high,” and “very high.” Other than the likelihood, an effect on the project (or impact) is assigned. These can be one of “insignificant,” “tolerable,” “serious,” and “catastrophic.” The risks are listed below and Table xx summarizes them.

1. The requirements change during the implementation.

    - Analysis: Since the software development life-cycle model is the Waterfall model, the requirements are set in this report and the likelihood of any change is not very high [18]. However, the technical domains of machine learning, web scraping and crawling are new to the Perfent engineering team, which means that if the engineers are stuck at some point, the requirements might have to be changed. The probability of this happening is moderate; therefore, the likelihood of this risk occurring is initially identified as moderate. The effect (or impact) on the project actually changes depending on how deep the team is into the implementation. If the changes occur in the early stages, the effect is insignificant since there is not much refactoring to do. However, if the changes occur in the late stages, the effect is serious since the existing code must be changed to welcome the new changes. Some components that were prepared to be used in the altered features may go to waste as well. Overall, the effect is identified as tolerable, assuming that the parts with existing code will not be changed too much.

    - Strategy: The best strategy is to analyze the requirements carefully and realistically in this report. But this is a difficult task, therefore the code must be developed in a way that welcomes changes. In addition, the requirements should be prioritized before the development. The features that have higher priorities

and are more essential to the core idea should be implemented before others since these are less likely to change. This way, new changes will not result in time and/or effort wastes and will be easier to implement.

2. The implementation time may be underestimated.

    - Analysis: Since the engineering team is not very experienced in the machine learning and web scraping domains, the likelihood of underestimating the implementation time is high. The effect on the project is tolerable since this is not an actual contracted project.
    - Strategy: The primary strategy is to research the new technologies beforehand to get a good idea of their difficulties. Creating throwaway prototypes to understand how they work also contributes to the strategy. This will be discussed in the next risk further. If the risk still occurs after these are done, some low-priority requirements might be removed from the project. This last strategy is not preferred, but if there is no other choice, it must be followed.
3. Tools and third-party services to be used during the implementation may not integrate well with the project.

    - Analysis: The project is not very big in size. Thanks to this, there will not be specialized tools and services that are difficult to integrate. The tools and services that are known to be able to be integrated can be used. For example: Google Calendar is a very popular calendar service and is known to be integrated to many projects. GitHub is similar in this sense but it is a tool. This follows that the likelihood of this risk occurring is low. In the case that it happens, the effect on the project is serious. If an important external component cannot be integrated when the team is deep into the implementation, we would have to look for alternatives. This would result in a waste of time when the deadlines are approaching.
    - Strategy: For all the external tools and services, the team will create a throwaway prototype to test how well they can be integrated. This way, we can understand if a tool or service can be integrated the way we want. In case one of these prototypes gives us false positive results, we will have replacement tools and services at the ready.
4. Technologies that are planned to be used may change.

    - Analysis: The technologies that we plan to use such as development frameworks like React and Spring, our choice of

web scraping tool, or the Google Calendar service may undergo changes that would render our plan outdated. The probability of such a situation is very low; however, the likes of it are known to happen. The effect of the risk is catastrophic. The team would have to refactor or move the entire project to some other technology.

- Strategy: We plan to avoid this risk by using popular technologies such as React and Spring. The popular technologies are less likely to undergo massive changes since that would affect many other projects. Even if such a case happens, these technologies do not force the developers to use the latest version.

5. A competitive product may be published before Perfent is.

    - Analysis: As discussed in the previous sections, there is no group event recommender system similar to Perfent currently. And since such an idea is not trending on the internet these days, we believe that the likelihood of this risk occurring is very low. The effect would be serious since the potential customers and investors would be drawn to the competitor's product since it came out first.
    - Strategy: In the case that the risk occurs, we will increase the amount of marketing and advertisements to get the customers' attention. In addition, we should identify the weaknesses of the competitor's product and implement what they are missing in our product to draw customers.

6. Engineers are ill or unable to continue working temporarily.

    - Analysis: Since the pandemic is not completely over yet and the development will mostly take place over winter where people are more vulnerable to both COVID-19 and other milder diseases; the probability of missing a group member temporarily is moderate. The effect on the project is serious since the parts that the missing engineer(s) are responsible for would not be advancing.
    - Strategy: In such a case, the work will be redistributed among the remaining engineers until those missing are able to come back. Normally, the lack of knowledge of other engineers' parts would be a problem. However, since we assigned at least two engineers to each part (parts: Front-end, web server, machine learning, web scraping), even if one member goes missing, there is at least one other person that is knowledgeable on the files and the code of the missing person. Thanks to this, a

potential redistribution of the tasks would not be a huge problem.

7. Project management may be insufficient in later stages.

    - Analysis: As people work on different parts of the project, the team communication and management may be weakened in the later stages. However, since we are a small team with constant discussions and collaborations, the probability is low. The effect is tolerable since if the team members notice a weak communication, they can call for a status meeting.
    - Strategy. To avoid this risk, we plan to have weekly status meetings. In these meetings, we plan to tell our fellow developers what we accomplished that week, what we will work on in the next week, and if there are any hindrances that keep us from moving on; similar to daily Scrums. We will also have task boards to let the team know our progress asynchronously.

8. The technical knowledge of the engineers may be insufficient.

    - Analysis: Since we are all new to machine learning and web scraping domains, the probability of this risk is high, even with thorough research on the subjects. The effect on the project is serious. After all, if we get stuck on a crucial part, we might have to unwillingly change the requirements, which is a hassle.
    - Strategy: Firstly, all the research will be done and small throwaway prototypes will be built for testing our knowledge. If there are still some parts that we find ourselves stuck in, we plan to consult our innovation expert who is very experienced in the recommendation system domain.

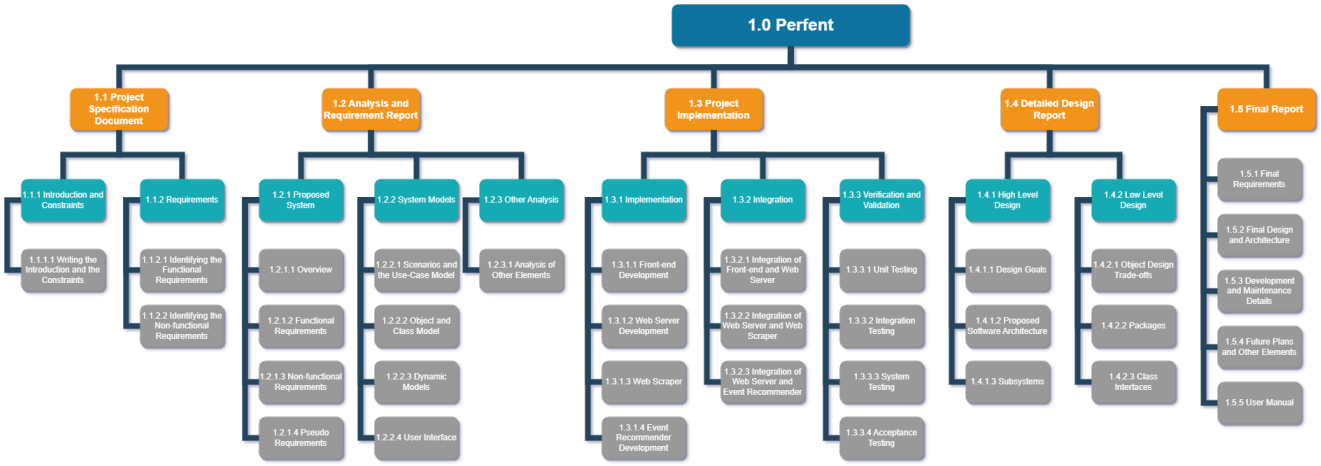
Table 1: Summary of the Risk Management Plan

Risk	Likelihood	Effect on the project	Strategy Summary
Requirement changes	Moderate	Tolerable	Develop the code in a way that allows flexibility. Prioritize the requirements and implement the higher-priority features before others.
Underestimation of the implementation time	High	Tolerable	Research the new technologies well before implementation to get a better understanding of the duration of tasks.
Tools and other services not integrating well	Low	Serious	Create throwaway prototypes to see how well the tools and services can be integrated to the project. Look for alternatives if any component cannot be integrated.
Technology changes	Low	Catastrophic	Use popular technologies to avoid the risk.
Competitors publishing products	Very low	Serious	Increase the amount of marketing and advertising. Identify the weaknesses of the competitor and provide what they cannot in Perfent.
Engineers not able to work	Moderate	Serious	Redistribute the tasks among the remaining engineers.
Insufficient project management	Low	Tolerable	Call weekly status meetings and use project management tools such as task boards.
Insufficient technical knowledge	High	Serious	Consult the innovation expert if research is not enough.

4.3 Project Plan

This section describes Perfent's project plan. A work breakdown structure (WBS) has been created (See Figure 36). We have a hybrid type of WBS. It is a combination of process-based and product-based WBSs. The top level is process based: The work is broken down as Specification, Analysis, Design, Implementation, Final Report. The Implementation process is broken down in a product-based fashion: Front-end, web server, web scraper, and event recommender.

Figure 36: The Work Breakdown Structure



For better readability, the left and the right parts of Figure 36 are presented separately below. The full image can be found in high resolution in this link.

Figure 37: The left part of the Work Breakdown Structure

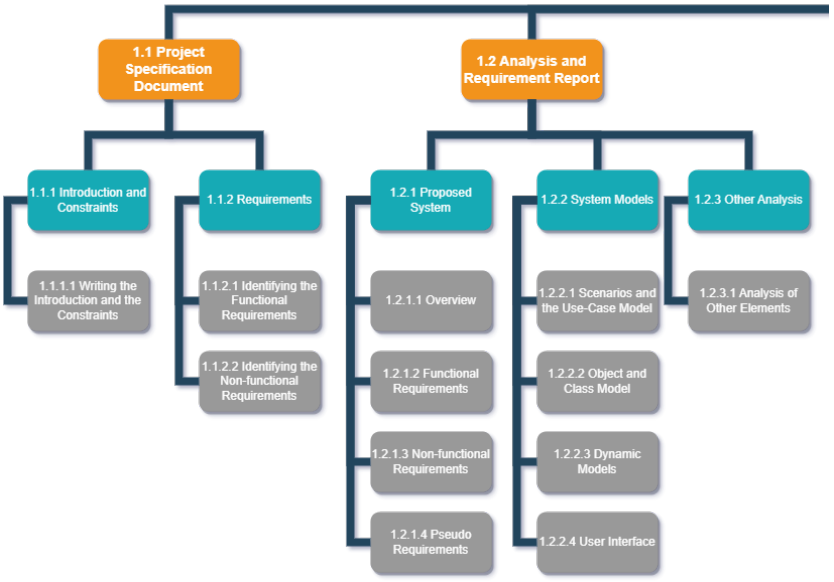
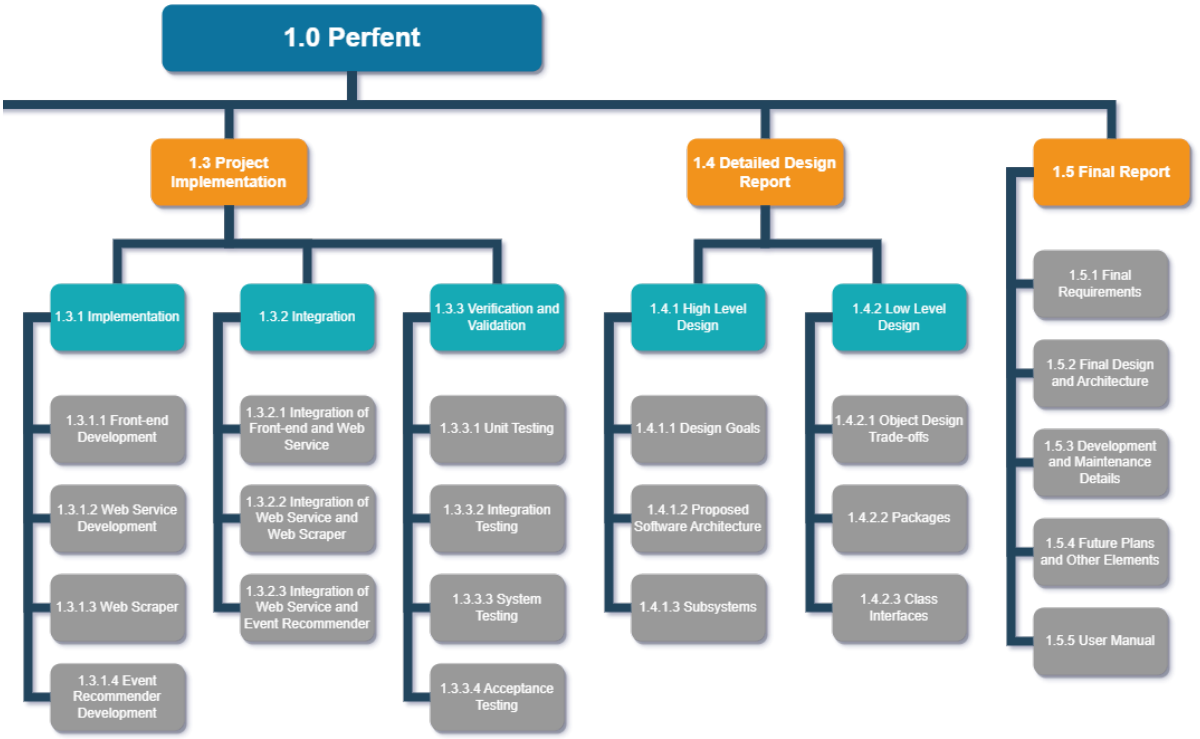


Figure 38: The right part of the Work Breakdown Structure



The work packages are listed in Table 2. The work package numbers (WP#) refer to the numbers in the WBS above. For each WP, the leader and the involved members are specified. Note that the leaders are also involved with the respective package, but their names are excluded from the “Members involved” lists for simplicity.

Table 2: List of work packages

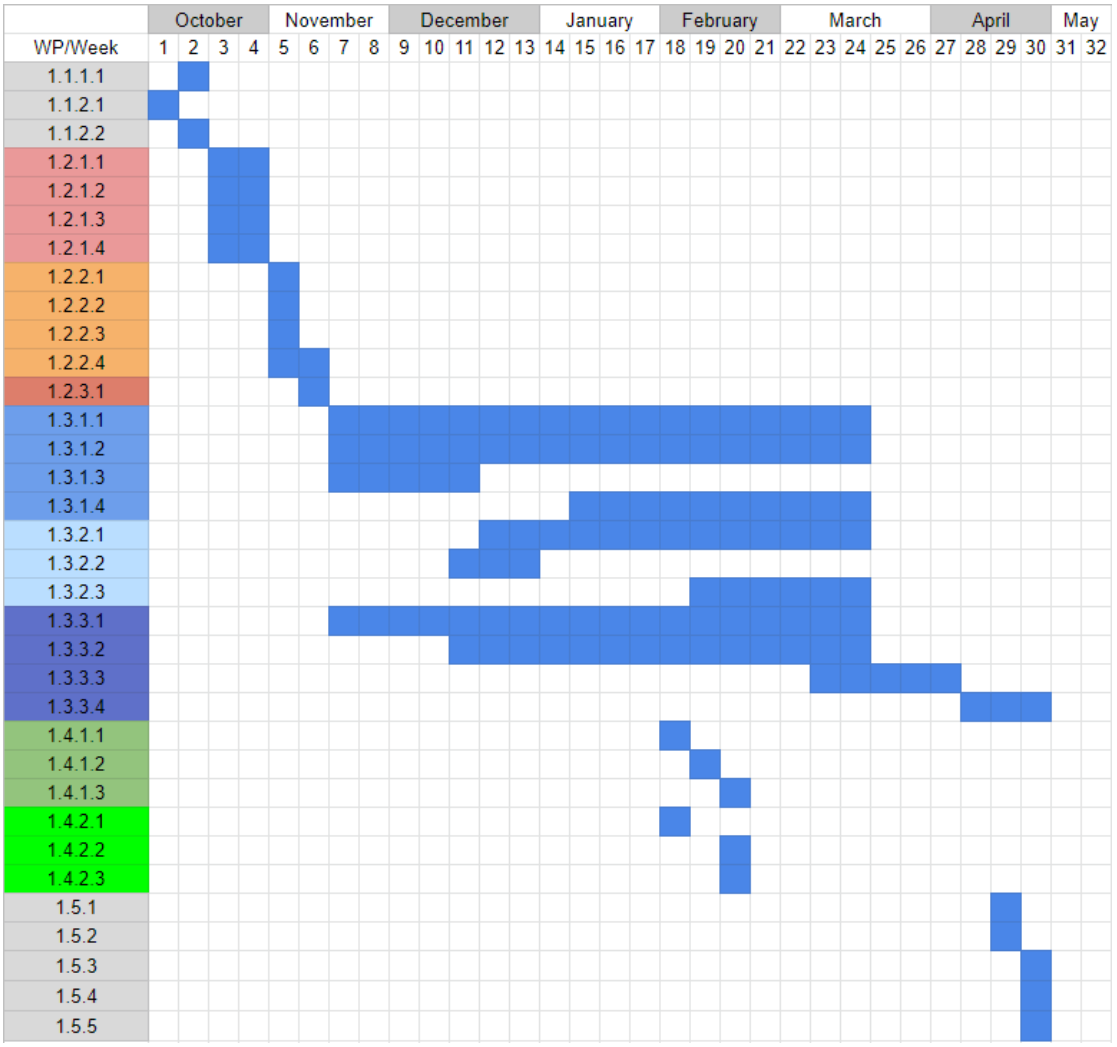
WP#	Work package title	Leader	Members involved
1.1.1.1	Writing the Introduction and the Constraints	Elif Çenesiz	Beste Güney
1.1.2.1	Identifying the Functional Requirements	Bora Çün	Çağrı Eren, Faruk Güney
1.1.2.2	Identifying the Non-functional Requirements	Çağrı Eren	Bora Çün, Faruk Güney
1.2.1.1	Overview	Bora Çün	
1.2.1.2	Functional Requirements	Çağrı Eren	Bora Çün
1.2.1.3	Non-functional Requirements	Çağrı Eren	
1.2.1.4	Pseudo Requirements	Beste Güney	

1.2.2.1	Scenarios and the Use-Case Model	Faruk Güney	Beste Güney, Çağrı Eren
1.2.2.2	Object and Class Model	Beste Güney	
1.2.2.3	Dynamic Models	Bora Çün	Çağrı Eren
1.2.2.4	User Interface	Elif Çenesiz	Faruk Güney
1.2.3.1	Analysis of Other Elements	Beste Güney	Bora Çün, Çağrı Eren, Elif Çenesiz
1.3.1.1	Front-end Development	Elif Çenesiz	Faruk Güney
1.3.1.2	Web Server Development	Beste Güney	Çağrı Eren, Bora Çün
1.3.1.3	Web Scraper	Bora Çün	Faruk Güney
1.3.1.4	Event Recommender Development	Çağrı Eren	Beste Güney, Elif Çenesiz
1.3.2.1	Integration of Front-end and Web Server	Beste Güney	Çağrı Eren, Bora Çün, Elif Çenesiz, Faruk Güney
1.3.2.2	Integration of Web Server and Web Scraper	Faruk Güney	Bora Çün, Beste Güney, Çağrı Eren
1.3.2.3	Integration of Web Server and Event Recommender	Çağrı Eren	Bora Çün, Beste Güney, Elif Çenesiz
1.3.3.1	Unit Testing	Faruk Güney	Beste Güney, Elif Çenesiz, Çağrı Eren, Bora Çün
1.3.3.2	Integration Testing	Beste Güney	Çağrı Eren
1.3.3.3	System Testing	Bora Çün	Beste Güney, Çağrı Eren, Elif Çenesiz, Faruk Güney
1.3.3.4	Acceptance Testing	Bora Çün	Beste Güney, Çağrı Eren, Elif Çenesiz, Faruk Güney
1.4.1.1	Design Goals	Faruk Güney	Beste Güney
1.4.1.2	Proposed Software Architecture	Elif Çenesiz	Bora Çün, Çağrı Eren
1.4.1.3	Subsystems	Faruk Güney	Beste Güney
1.4.2.1	Object Design Trade-offs	Çağrı Eren	
1.4.2.2	Packages	Elif Çenesiz	Bora Çün, Faruk Güney

1.4.2.3	Class Interfaces	Beste Güney	Çağrı Eren, Elif Çenesiz, Faruk Güney, Bora Çün
1.5.1	Final Requirements	Faruk Güney	Bora Çün
1.5.2	Final Design and Architecture	Elif Çenesiz	Beste Güney
1.5.3	Development and Maintenance Details	Çağrı Eren	Beste Güney, Faruk Güney
1.5.4	Future Plans and Other Elements	Elif Çenesiz	Faruk Güney
1.5.5	User Manual	Bora Çün	Çağrı Eren

The initial estimations for the duration of these work packages are presented below using a Gantt Chart. The Gantt Chart does not have the usual “waterfall” shape observed in most Gantt Charts. This is because the work packages are not ordered in chronological order. They are ordered in the order they appear in the WBS. Please note that the WP durations are initial estimates and are subject to change depending on the project status.

Figure 39: Project Gantt Chart



The rest of this subsection consists of the WP tables. Each table explains the content of the WPs in detail. The WP numbers refer to the numbers in the WBS.

WP 1.1.1.1: *Writing the Introduction and the Constraints*			
Start date: *10.10.2022* **End date:** *16.10.2022*			
Leader:	*Elif Çenesiz*	**Members involved:**	*Beste Güney*
Objectives: *An introduction section will be created to introduce the reader to the Specification Document. Perfent's aims and the problems it solves will be explained. The project constraints will be discussed in detail.*			
Tasks: ***Task 1.1.1.1.1 Writing the Introduction*:** *The introduction will be written to explain Perfent's aims and the problems it solves.* ***Task 1.1.1.1.2 Identifying the project's constraints*:** *The project's constraints will be identified and the corresponding document section will be written.*			
Deliverables ***D1.1.1.1.1 Project Specification Document***			
WP 1.1.2.1: *Identifying the Functional Requirements*			
Start date: *03.10.2022* **End date:** *09.10.2022*			
Leader:	*Bora Çün*	**Members involved:**	*Çağrı Eren, Faruk Güney*
Objectives: *Identifying the functional requirements of the project. Categorizing the different functionalities. Completing the relevant subsection in the document.*			
Tasks: ***Task 1.1.2.1.1 Identify the feature categories*:** *The different categories of features such as schedule features and event features will be identified.* ***Task 1.1.2.1.2 Explain the functional requirements*:** *The functional requirements will be written down. These requirements will be sufficiently explained.* ***Task 1.1.2.1.3 Complete the Subsection*:** *The subsection containing the functional requirements will be completed. The necessary overview texts will be written.*			
Deliverables			

D1.1.2.1.1 Project Specification Document

WP 1.1.2.2: *Identifying the Non-functional Requirements*

Start date: *10.10.2022* **End date:** *16.10.2022*

Leader:	*Çağrı Eren*	**Members involved:**	*Bora Çün, Faruk Güney*

Objectives: *Identifying the non-functional requirements of the project. Categorizing these. Completing the relevant subsection in the document.*

Tasks:

Task 1.1.2.2.1 Identify the categories**:** *The different categories of non-functional requirements such as performance and usability will be identified.*

Task 1.1.2.2.2 Explain the non-functional requirements**:** *The non-functional requirements will be written down. These requirements will be sufficiently explained.*

Task 1.1.2.2.3 Complete the Subsection**:** *The subsection containing the non-functional requirements will be completed.*

Deliverables

D1.1.2.2.1 Project Specification Document

WP 1.2.1.1: *Overview*

Start date: *17.10.2022* **End date:** *30.10.2022*

Leader:	*Bora Çün*	**Members involved:**	

Objectives: *Providing an overview for the project requirements. Giving the reader an idea about the project before reading the requirements. Clarifying the points that are difficult to explain with bullet lists.*

Tasks:

Task 1.2.1.1.1 Write the overview text: *The overview text will be structured and written.*

Deliverables

D1.2.1.1.1 Analysis and Requirement Report

WP 1.2.1.2: *Functional Requirements*

Start date: *17.10.2022* **End date:** *30.10.2022*

Leader:	*Çağrı Eren*	**Members involved:**	*Bora Çün*

Objectives: *Moving the functional requirements from the Specification Document to the Analysis and Requirement Report. Making the necessary adjustments to the currently existing requirements. Providing a better readability.*

Tasks:

Task 1.2.1.2.1 Move the functional requirements: *Move the functional requirements to the Analysis and Requirement Report.*

Task 1.2.1.2.2 Make the necessary adjustments: *Make the necessary adjustments based on supervisor and advisor feedback.*

Deliverables

D1.2.1.2.1 Analysis and Requirement Report

WP 1.2.1.3: *Non-functional Requirements*

Start date: *17.10.2022* **End date:** *30.10.2022*

Leader:	*Çağrı Eren*	**Members involved:**	

Objectives: *Moving the non-functional requirements from the Specification Document to the Analysis and Requirement Report. Making the necessary adjustments to the currently existing requirements. Providing a better readability.*

Tasks:

Task 1.2.1.3.1 Move the non-functional requirements: *Move the non-functional requirements to the Analysis and Requirement Report.*

Task 1.2.1.3.2 Make the necessary adjustments: *Make the necessary adjustments based on supervisor and advisor feedback.*

Deliverables

D1.2.1.3.1 Analysis and Requirement Report

WP 1.2.1.4: *Pseudo Requirements*

Start date: *17.10.2022* **End date:** *30.10.2022*

Leader:	*Beste Güney*	**Members involved:**	

Objectives: *Moving the implementation constraints from the Specification Document to the Analysis and Requirement Report. Enhancing the existing ones with more technical information. Providing a better readability.*

Tasks:

Task 1.2.1.4.1 Move the constraints: *Move the constraints to the Analysis and Requirement Report. Convert these to pseudo requirements.*

Task 1.2.1.4.2 Enhance the pseudo requirements: *Enhancing the existing requirements with more technical information.*

Deliverables

D1.2.1.4.1 Analysis and Requirement Report

WP 1.2.2.1: *Scenarios and the Use-Case Model*

Start date: *31.10.2022* **End date:** *06.11.2022*

Leader:	*Faruk Güney*	**Members involved:**	*Beste Güney, Çağrı Eren*

Objectives: *Determining the various scenarios that the users will be in when using Perfent. Converting the existing functional requirement to use cases. Identifying the actors involved. Drawing the Use-Case Diagram.*

Tasks:

Task 1.2.2.1.1 Determine the scenarios: *Determine the various scenarios that the users will be in when using Perfent.*

Task 1.2.2.1.2 Identify the use-cases: *Converting the existing functional requirement to use cases.*

Task 1.2.2.1.3 Identify the actors: *Identify the actors involved in the use-case diagram.*

Task 1.2.2.1.4 Draw the Use-Case Diagram: *Draw the Use-Case Diagram using the use-cases and the actors.*

Deliverables

D1.2.2.1.1 Analysis and Requirement Report

WP 1.2.2.2: *Object and Class Model*

Start date: *31.10.2022* **End date:** *06.11.2022*

Leader:	*Beste Güney*	**Members involved:**	

Objectives: Determining the classes that will be needed in the software. Identifying the relations between the objects. Completing the Class Diagram of Perfent software.

Tasks:

Task 1.2.2.2.1 Determine the classes: *Determine the classes and the object relations.*

Task 1.2.2.2.2 Draw the Class Diagram: *Complete the Class Diagram.*

Deliverables

D1.2.2.2.1 Analysis and Requirement Report

WP 1.2.2.3: *Dynamic Models*

Start date: *31.10.2022* **End date:** *06.11.2022*

Leader:	*Bora Çün*	**Members involved:**	*Çağrı Eren*

Objectives: Producing descriptions of some non-trivial processes using Dynamic Models. These Dynamic Models include State Diagrams, Activity Diagrams, and Sequence Diagrams.

Tasks:

Task 1.2.2.3.1 Identify the processes: *The processes where a dynamic model needs to be drawn will be identified.*

Task 1.2.2.3.2 Determine a proper diagram for each process: *For each process that was identified in the previous task, the type of diagram will be determined.*

Task 1.2.2.3.3 Draw the diagrams: *The diagrams will be drawn and explanations will be written.*

Deliverables

D1.2.2.3.1 Analysis and Requirement Report

WP 1.2.2.4: *User Interface*

Start date: *31.10.2022* **End date:** *13.11.2022*

Leader:	*Elif Çenesiz*	**Members involved:**	*Faruk Güney*

Objectives: To create visual mock-ups of how Perfent will look like when the production is over. The UI mock-ups will give the reader a visual explanation of the app. The outcomes of this WP will act as a guide for the front-end development.

Tasks:

Task 1.2.2.4.1 Identify the screens to be drawn: *Identify the screens to be drawn in this subsection.*

Task 1.2.2.4.2 Draw the screens: *The screen mock-ups will be drawn using Figma [19].*

Task 1.2.2.4.3 Explain the UI figures: *Each non-trivial screen mock-up produced will be explained.*

Deliverables

D1.2.2.4.1 Analysis and Requirement Report

WP 1.2.3.1: *Analysis of Other Elements*

Start date: *07.11.2022* **End date:** *13.11.2022*

Leader:	*Beste Güney*	**Members involved:**	*Bora Çün, Çağrı Eren, Elif Çenesiz*

Objectives: *The rest of the analysis will be explained. After reading the report, the reader should be informed about Perfent's consideration of various factors, risk management strategy, project plan, teamwork, professional responsibilities, and learning strategies among other things.*

Tasks:

Task 1.2.3.1.1 Determine the various factors in engineering design: *The various factors in engineering design will be determined and explained.*

Task 1.2.3.1.2 Analyzing the risks: *The initial risks will be analyzed and proper strategies will be discussed.*

Task 1.2.3.1.3 Come up with a project plan: *Estimate a realistic project plan. Create the WBS and explain the WPs. Draw a Gantt Chart to help visualize.*

Task 1.2.3.1.4 Explain the teamwork: *Explain how Perfent engineers will ensure an efficient teamwork.*

Task 1.2.3.1.5 Discuss ethics and professional responsibilities: *Discuss ethics and professional responsibilities related to Perfent.*

Task 1.2.3.1.6 Explain the learning strategy: *Explain how the team members plan for new knowledge. Discuss the learning strategy.*

Deliverables

D1.2.2.4.1 Analysis and Requirement Report

WP 1.3.1.1: *Front-end Development*

Start date: *14.11.2022* **End date:** *19.03.2023*

Leader:	*Elif Çenesiz*	**Members involved:**	*Faruk Güney*

Objectives: *To implement the user interface of the application. The UI should be interacting with the web server. The user experience should be inline with the requirements.*

Tasks:

Task 1.3.1.1.1 Set the front-end framework up: *The front-end development environment will be set up.*

Task 1.3.1.1.2 Implement the static pages: *The static versions of the pages with no functionalities will be implemented.*

Task 1.3.1.1.3 Add functionality to the pages: *Add functionality to the pages to allow the user to interact with the app.*

Deliverables

D1.3.1.1.1 Implementation Code

WP 1.3.1.2: *Web Server Development*

Start date: *14.11.2022* **End date:** *19.03.2023*

Leader:	*Beste Güney*	**Members involved:**	*Çağrı Eren, Bora Çün*

Objectives: *To implement a backend service that the front-end can send requests to and receive responses from. The web server must have the necessary operations to support the front-end. The web server must be integrated well with both the web scraper and the event recommender app.*

Tasks:

Task 1.3.1.2.1 Set the web server framework up: *The web server development environment will be set up.*

Task 1.3.1.2.2 Implement the business logic: *The business logic of the application will be implemented.*

Task 1.3.1.2.3 Implement the API endpoints: *Implement the API endpoints that the front-end will be interacting with.*

Deliverables

D1.3.1.2.1 Implementation Code

WP 1.3.1.3: *Web Scraper*

Start date: *14.11.2022* **End date:** *18.12.2022*

Leader:	*Bora Çün*	**Members involved:**	*Faruk Güney*

Objectives: *To implement a web scraper application. The web scraper implemented here will be used to get event data from the internet. This application should be integrable with the web server.*
Tasks: ***Task 1.3.1.3.1 Set the web scraper framework up:*** *The web scraper development environment will be set up.* ***Task 1.3.1.3.2 Implement the web scraper:*** *The web scraper application will be implemented.* ***Task 1.3.1.3.3 Determine the event websites:*** *The event websites from where the event data will be gathered will be determined.*
Deliverables ***D1.3.1.3.1 Implementation Code***

WP 1.3.1.4: *Event Recommender Development*			
Start date: *09.01.2023* **End date:** *19.03.2023*			
Leader:	*Çağrı Eren*	**Members involved:**	*Beste Güney, Elif Çenesiz*
Objectives: *To implement a machine learning model for recommending events. The model should recommend events that make sense. The model should be integrable with the web server.*			
Tasks: ***Task 1.3.1.4.1 Create the model:*** *Create the machine learning model for recommending events.*			
Deliverables ***D1.3.1.4.1 Implementation Code***			

WP 1.3.2.1: *Integration of Front-end and Web Server*			
Start date: *19.12.2022* **End date:** *19.03.2023*			
Leader:	*Beste Güney*	**Members involved:**	*Çağrı Eren, Bora Çün, Elif Çenesiz, Faruk Güney*

Objectives: *To have Perfent's front-end and web server applications connected to each other. At the end of this activity, the front-end should be communicating with the web server without problems. The user should not see any hard coded information from the front-end.*
Tasks: ***Task 1.3.2.1.1 Connect front-end functionality to web server endpoints:*** *Send requests to the web server when the front-end needs to read or write data. Receive requests and behave accordingly.*
Deliverables ***D1.3.2.1.1 Implementation Code***

WP 1.3.2.2: *Integration of Web Server and Web Scraper*			
Start date: *12.12.2022* **End date:** *01.01.2023*			
Leader:	*Faruk Güney*	**Members involved:**	*Bora Çün, Beste Güney, Çağrı Eren*
Objectives: *To have Perfent's web scraper and web server applications connected to each other. At the end of this activity, the web scraper should be sending the event data it finds to the web server and the database. The web server should not be creating event data on its own, these should come from the web scraper.*			
Tasks: ***Task 1.3.2.1.1 Connect the web scraper to the web server:*** *Send requests to the web server about writing new event data.*			
Deliverables ***D1.3.2.2.1 Implementation Code***			

WP 1.3.2.3: *Integration of Web Server and Event Recommender*			
Start date: *06.02.2023* **End date:** *19.03.2023*			
Leader:	*Çağrı Eren*	**Members involved:**	*Bora Çün, Beste Güney, Elif Çenesiz*
Objectives: *To have Perfent's event recommender and web server applications connected to each other. At the end of this activity, the web*			

server should be able to reach the event recommendations that are prepared by the event recommender.
Tasks: ***Task 1.3.2.1.1 Connect the event recommender to the web server:*** *Create a communication channel where web server can reach the event recommendations of the event recommender.*
Deliverables ***D1.3.2.3.1 Implementation Code***

WP 1.3.3.1: *Unit Testing*			
Start date: *14.11.2022* **End date:** *19.03.2023*			
Leader:	*Faruk Güney*	**Members involved:**	*Beste Güney, Elif Çenesiz, Çağrı Eren, Bora Çün*
Objectives: *To have all the basic components of the applications tested. Each unit should be behaving correctly when compared to the functional requirements. The engineers should not be pushing commits to the repository without having all the unit tests passing.*			
Tasks: ***Task 1.3.3.1.1 Set the unit testing environment up:*** *A unit test writing environment should be set up so that the engineers can write and execute tests.* ***Task 1.3.3.1.2 Write unit tests for the basic components:*** *Each basic component should have a unit test written. No code should be pushed without all the tests passing.*			
Deliverables ***D1.3.3.1.1 Test code***			

WP 1.3.3.2: *Integration Testing*			
Start date: *12.12.2022* **End date:** *19.03.2023*			
Leader:	*Beste Güney*	**Members involved:**	*Çağrı Eren*

Objectives: *To have all the integration activities verified. The components should be interacting with each other correctly when compared to the requirements. The engineers should not be pushing integration commits to the repository without verifying the correctness.*
Tasks: ***Task 1.3.3.2.1 Set the necessary scaffoldings:*** *The necessary scaffoldings will be set up to allow testing.* ***Task 1.3.3.2.2 Verify each integration activity:*** *Test and compare against the interface specifications*
Deliverables ***D1.3.3.2.1 Test code***

WP 1.3.3.3: *System Testing*			
Start date: *06.03.2023* **End date:** *09.04.2023*			
Leader:	*Bora Çün*	**Members involved:**	*Beste Güney, Çağrı Eren, Elif Çenesiz, Faruk Güney*
Objectives: *To have the product at hand verified. The entire system should be tested against the requirements stated in this report. Any inconsistencies with the requirements should be addressed.*			
Tasks: ***Task 1.3.3.3.1 Test each requirement in the product:*** *Each functional and non-functional requirement will be tested in the current state of the product.*			
Deliverables ***D1.3.3.3.1 System Test Report***			

WP 1.3.3.4: *Acceptance Testing*			
Start date: *10.04.2023* **End date:** *30.04.2023*			
Leader:	*Bora Çün*	**Members involved:**	*Beste Güney, Çağrı Eren, Elif Çenesiz, Faruk Güney*

Objectives: *To have the product at hand validated. The entire system should be tested for usefulness and user satisfaction.*
Tasks: ***Task 1.3.3.4.1 Find external users:*** *Find users other than the engineers to help in the acceptance testing.* ***Task 1.3.3.4.2 Create test scenarios:*** *Determine some scenarios that the testers can validate.* ***Task 1.3.3.4.3 Have the testers validate the product:*** *Ask the testers to complete the tasks in the scenarios. Ask the testers to roam freely in the application independent of the scenarios. Ask their opinions on the user satisfaction.*
Deliverables ***D1.3.3.4.1 Acceptance Test Document***

WP 1.4.1.1: *Design Goals*			
Start date: *30.01.2023* **End date:** *05.02.2023*			
Leader:	*Faruk Güney*	**Members involved:**	*Beste Güney*
Objectives: *To identify the design goals in the software. This section should contain sufficient explanation for each goal. The reader should be well-informed after reading.*			
Tasks: ***Task 1.4.1.1.1 Identify the design goals:*** *Identify the design goals of Perfent.*			
Deliverables ***D1.4.1.1.1 Detailed Design Report***			

WP 1.4.1.2: *Proposed Software Architecture*			
Start date: *06.02.2023* **End date:** *12.02.2023*			
Leader:	*Elif Çenesiz*	**Members involved:**	*Bora Çün, Çağrı Eren*

Objectives: *To determine a software architecture that satisfies the design goal described in the previous section. The architecture should be inline with the pseudo requirements stated in this report.*

Tasks:

Task 1.4.1.2.1 Identify the best software architecture: *Identify the best software architecture that satisfies the design goals and the pseudo requirements of Perfent.*

Task 1.4.1.2.2 Express the architecture: *Express the architecture as a UML diagram and explain in text.*

Deliverables

D1.4.1.2.1 Detailed Design Report

WP 1.4.1.3: *Subsystems*

Start date: *13.02.2023* **End date:** *19.02.2023*

Leader:	*Faruk Güney*	**Members involved:**	*Beste Güney*

Objectives: *To identify the necessary subsystems for the development of Perfent. These subsystems should be consistent with Perfent's software architecture.*

Tasks:

Task 1.4.1.3.1 Identify the subsystems: *Identify the necessary subsystems for the development of Perfent.*

Task 1.4.1.3.2 Express the subsystems: *Express the subsystems as a UML diagram and explain in text.*

Deliverables

D1.4.1.3.1 Detailed Design Report

WP 1.4.2.1: *Object Design Trade-offs*

Start date: *30.01.2023* **End date:** *05.02.2023*

Leader:	*Çağrı Eren*	**Members involved:**	

Objectives: *To explain the design choices by explaining the trade-offs. Each trade-off should be identified and explained in detail.*
Tasks: ***Task 1.4.2.1.1 Identify and explain the trade-offs:*** *Identify and explain the object design trade-offs.*
Deliverables ***D1.4.2.1.1 Detailed Design Report***

WP 1.4.2.2: *Packages*			
Start date: *13.02.2023* **End date:** *19.02.2023*			
Leader:	*Elif Çenesiz*	**Members involved:**	*Bora Çün, Faruk Güney*
Objectives: *To identify the necessary packages for the development of Perfent. These packages should be consistent with Perfent's software architecture and subsystems.*			
Tasks: ***Task 1.4.2.2.1 Identify the packages:*** *Identify the necessary packages for the development of Perfent.* ***Task 1.4.2.2.2 Express the subsystems:*** *Express the packages as a UML diagram and explain in text.*			
Deliverables ***D1.4.2.2.1 Detailed Design Report***			

WP 1.4.2.3: *Class Interfaces*			
Start date: *13.02.2023* **End date:** *19.02.2023*			
Leader:	*Beste Güney*	**Members involved:**	*Çağrı Eren, Elif Çenesiz, Faruk Güney, Bora Çün*
Objectives: *To express the class interfaces as tables. The reader should be able to understand the interfaces easily.*			
Tasks:			

Task 1.4.2.3.1 Convert the class interfaces to tables: *Convert and explain the class interfaces in tables.*
Deliverables ***D1.4.2.3.1 Detailed Design Report***

WP 1.5.1: *Final Requirements*			
Start date: *17.04.2023* **End date:** *23.04.2023*			
Leader:	*Faruk Güney*	**Members involved:**	*Bora Çün*
Objectives: *To present the final versions of the requirements. At this point, the project must be either completed or very close to completion; which means that there should not be any requirement changes.*			
Tasks: ***Task 1.5.1.1 List the final versions of the requirements:*** *List the final versions of the requirements and explain each one.*			
Deliverables ***D1.5.1.1 Final Report***			

WP 1.5.2: *Final Design and Architecture*			
Start date: *17.04.2023* **End date:** *23.04.2023*			
Leader:	*Elif Çenesiz*	**Members involved:**	*Beste Güney*
Objectives: *To present the final versions of the design and architecture. At this point, the project must be either completed or very close to completion; which means that there should not be any design or architecture changes.*			
Tasks: ***Task 1.5.2.1 Present the final versions of the design and architecture:*** *Present the final versions of the design and architecture in UML and text.*			
Deliverables ***D1.5.2.1 Final Report***			

WP 1.5.3: *Development and Maintenance Details*			
Start date: *24.04.2023* **End date:** *30.04.2023*			
Leader:	*Çağrı Eren*	**Members involved:**	*Beste Güney, Faruk Güney*
Objectives: *To explain how the development was carried out. If there were any differences with what was stated in this report and what actually happened, these should be stated and explained. If future maintenance is to be done, this should be explained in detail.*			
Tasks: ***Task 1.5.3.1 Explain the development:*** *Explain how the development was carried out.* ***Task 1.5.3.2 State the differences:*** *State any differences with what was stated in this report and what actually happened. Explain the reasons for the changes.* ***Task 1.5.3.3 Explain future maintenance:*** *If future maintenance is to be done, this should be explained in detail.*			
Deliverables ***D1.5.3.1 Final Report***			

WP 1.5.4: *Future Plans and Other Elements*			
Start date: *24.04.2023* **End date:** *30.04.2023*			
Leader:	*Elif Çenesiz*	**Members involved:**	*Faruk Güney*
Objectives: *To inform the reader about Perfent's future after the CS fair. To inform the reader about how the analysis elements from this report were handled throughout the project.*			
Tasks: ***Task 1.5.4.1 Explain the future plans:*** *Explain the future plans.* ***Task 1.5.3.2 Explain the analysis elements:*** *Explain how the analysis elements were handled throughout the project.*			
Deliverables			

D1.5.4.1 Final Report

WP 1.5.5: *User Manual*			
Start date: *24.04.2023* **End date:** *30.04.2023*			
Leader:	*Bora Çün*	**Members involved:**	*Çağrı Eren*
Objectives: *To produce a user manual for Perfent. The manual should be easy to understand. It should cover the application comprehensively. Especially the tester questions from the acceptance testing should be answered.*			
Tasks: ***Task 1.5.5.1 Write the user manual:*** *Write a comprehensive and easily understandable user manual for the whole application.*			
Deliverables ***D1.5.5.1 Final Report***			

4.4 Ensuring Proper Teamwork

Perfent is a team project and the Perfent engineers realize that the project cannot reach its maximum potential without proper teamwork. To ensure proper teamwork, we have agreed to take the following measures:

- Equal work distribution: To avoid a potential imbalance in the work distribution, we will try to distribute the work as equally as possible. If some members have other external workload, the other members can take some of that member's work to help them.
- Weekly Status Meetings: As mentioned in the Risks and Alternatives section, we plan to have weekly status meetings. In these meetings, we plan to tell our fellow developers what we accomplished that week, what we will work on in the next week, and if there are any hindrances that keep us from moving on; similar to daily Scrums.
- Maintaining a Kanban Board: We plan to have a kanban board which will be updated as we complete more work [20]. The purpose of this board is to visualize the completed and remaining work amounts, which will lead to more motivated team members as well as more transparency between the team members.

- Assigning team members to tasks: To avoid the diffusion of responsibility, we will assign a team member to each issue and task.
- Code Reviews: To decrease the amount of defective code pushed to the repository, we will work with pull requests and review code before merging new code to the main branch. This will also help establish collective ownership of the code.

4.5 Ethics and Professional Responsibilities

The schedule information provided by the users, or any other confidential information will not be shared with any third party company unless the user agrees to share. Sensitive user information such as user passwords and locations must be stored securely. To achieve this, the data will be encrypted or hashed before they are uploaded to the database.

The Perfent team wants to work in a positive working environment. To achieve this, the developers will respect each other, the work will be shared as evenly as possible, and the developers will be transparent with their progress during the weekly progress meetings.

A user can report another user in case of an inappropriate situation.

4.6 Planning for New Knowledge and Learning Strategies

Knowledge is the most fundamental part of implementing any project since any type of implementation requires knowledge to implement the system. Therefore, implementers of a system need to determine strategies to learn that knowledge. This situation remains the same for the Perfent as there are many parts of the project implementers have little pre-existing knowledge and implementers need to gather knowledge about implementing those parts. In the following, we will see how Perfent's implementers acquire knowledge of the main parts of the project.

4.6.1 Acquiring Knowledge for Implementing Web Server

For the implementation of the Perfent backend web server, the Perfent team decided to use the Spring Boot framework since the team already has previous experience using the framework and working with web servers. This situation removes the process of watching long tutorials to learn how to code with Spring Boot starting from zero knowledge. Therefore, the Perfent team is well versed in setting up endpoints, writing business functionality, doing user authentication, and creating entity objects by using Spring Boot functionalities.

However, during implementation, there will be places where best-practice approaches need to be researched to implement something or

there is a bug that is emerging because the team has lack of knowledge using Spring Boot. Therefore, to acquire knowledge to solve such issues Perfent team plan to use Spring documentation [21] and technical forum sites like StackOverflow [22] and sites that offer technical knowledge like TutoriolsPoint[23].

4.6.2 Acquiring Knowledge for Implementing Frontend

For the implementation of the Perfent frontend Perfent team decided to use React since the team already has previous experience using React. Again this situation removes the process of acquiring knowledge to learn React from zero knowledge.

However, during implementation, we might use other libraries that we have not used before which provide functionalities that work in React since there are lots of useful libraries that work in React. Such libraries generally have their own respective documentation that can be used to acquire knowledge about those libraries and learn how the library is used. We plan to read such documentation when we are using libraries that work in React.

Again, during implementation, there will be places where the best practice approach needs to be researched to implement something or there is a bug that is emerging because the team has a lack of knowledge using React. Therefore, to acquire knowledge to solve such issues Perfent team plan to use official React documentation [24], technical forum sites like StackOverflow [22], and sites that offer technical knowledge like TutoriolsPoint [23].

4.6.3 Acquiring Knowledge for Implementing Web Scraper

The web scraping part of Perfent is the part that the Perfent team has no pre-existing experience. Therefore, this part requires an excessive amount of research and knowledge gathering.

First, we plan to research the popular approaches and best practices that are used in web scraping programs or there might be frameworks that specialize in web scraping. To discover the existence of such things and acquire sufficient knowledge we plan to use tutorials on YouTube [25], other tutorial sites such as TutorialsPoint [23], StackOverflow [22] answers that are about web scraping’s best practices and most importantly Google Search [26] engine as discovering as just entering search queries like “Web scraping frameworks”, “What are the best practices at web scraping”, or “Web scraping for beginners” are really beneficial for learning the existence of information.

Secondly, when we are done with acquiring the knowledge of what techniques we should use while doing web scraping, we will move on to how those techniques work and what the best practices for using those techniques are. To gather such knowledge, we plan to mainly use specific tutorials about those techniques on YouTube [25] as YouTube tutorials are really nice to learn things when one has no or limited knowledge. We also plan to use tutorial sites such as TutoriolsPoint [23] and StackOverflow[22] for acquiring knowledge about specific areas of that specific technique.

Finally, when we are well versed about what technique to use and how to use it only implementation remains. During the implementation, we will face problems where we are not sure of the best practice and there are bugs emerging because of our lack of knowledge about some specific part of the technique we are using. To tackle such problems we will mainly use technical forum sites like StackOverflow [22], sites that offer technical knowledge like TutoriolsPoint [23], and the official documentation of the technique we are using if it exists to acquire knowledge.

4.6.4 Acquiring Knowledge for Implementing Recommendation System

The recommendation system of the Perfent is one of the most important parts of the Perfent and one of the parts that the Perfent team has limited knowledge of. Therefore, an excessive amount of research about recommendation systems and machine learning is needed. We have already acquired knowledge about how to gather user data during our meeting with the innovation expert since our innovation expert is very knowledgeable about recommendation systems and machine learning.

First, although one of the team members finished the machine learning course last semester and one of the team members is taking it right now, as a team we want to go through a tutorial on how machine learning basics and recommendations systems work. We mainly plan to use YouTube [25] to acquire the knowledge for this part since YouTube offers good tutorials for beginners.

Secondly, we will specifically research the existence of techniques and libraries that are used in recommendation systems and if we can find specifically libraries and techniques used for group recommendations instead of single person recommendations. Luckily for us, our supervisor instructor and innovation expert have experience working on recommendation systems. Therefore, our primary resource to acquire knowledge about the existing techniques and libraries we can use for group and solo recommendations will be them. We will also use the Google Search engine [26] to acquire knowledge on the existence of such techniques and libraries. Another very important resource we will use will be academic research papers as there is

countless research about machine learning and recommendation systems since it is a really popular subject in computer science. Determining recommendations for a group of people might be niche and we might have a hard time finding resources in the Google Search engine although there are research papers about group recommendations. Therefore we will also use Google Scholar[27] to find and acquire knowledge from research papers.

Thirdly, when we are done with the research of the techniques and libraries and we have determined what to use, we will move on to acquiring knowledge about how we can implement such a system using the techniques and libraries we have determined to use. We will mainly use YouTube [25] tutorials to acquire knowledge since we will probably have little idea about the implementation of the technique we are using and YouTube is great for beginners to learn things. We will also use other websites that might help us at acquiring the knowledge of how we implement such a system like TutorialsPoint [23].

Finally, during the implementation stage of the recommendation systems, we will face problems such as bugs emerging because of our lack of knowledge about how to implement such a system and we are not sure what is the best approach to a section we are trying to implement. To tackle such specific issues we will mainly use technical data science forum sites like Data Science StackExchange [28] and StackOverflow [22], sites that offer technical knowledge like TutorialsPoint [23], and the official documentation of the technique we are using if it exists to acquire knowledge.

4.6.5 Acquiring Knowledge for Setting Up AWS Server

As of writing, this report Perfent's AWS Server setup is already finished. Our AWS Server consists of an Amazon Linux EC2 virtual machine instance with CI/CD pipeline connected to Perfent's GitHub repositories and DNS setup connected to the perfent.net domain.

The person who created the Amazon Linux instance already had pre-existing knowledge about creating an instance at AWS, therefore this part did not require gathering completely new knowledge but it did require reading AWS EC2 documentation [29] to gather specific information about specific parts of the creation process.

Setting up DNS for our AWS EC2 instance and connecting the perfent.net domain we bought from Google Domains was an area in which we had no pre-existing knowledge. Therefore, the person who is working on this part has mainly used AWS documentation about setting up DNS [30] to

acquire new knowledge and to acquire knowledge about specific issues StackOverflow [22] answers about setting up DNS for AWS are used.

The hardest part was setting up the CI/CD pipeline between the AWS EC2 instance and Perfent Github repositories since this part was not straightforward as others. Luckily for us, there was a perfect YouTube tutorial [31] showing the whole process. The person who is creating the pipeline mainly used this tutorial to gather knowledge about the setup. After using that resource again related StackOverflow answers [22] are used to acquire knowledge about specific issues.

5 Glossary

Proposing an event: A group member finds an event and suggests that the group attend that event.

Event artist: An important person in the event whether events are based on them or they are an integral part of it like the singer at a concert or the artist of the art gallery in an art gallery tour event.

Event runner: Event organizer who has registered and opened a user in the Perfent system.

6 References

[1] “We are what we do,” *Meetup*. [Online]. Available: https://www.meetup.com/. [Accessed: 16-Oct-2022].
[2] *Eventbrite*. [Online]. Available: https://www.eventbrite.com/. [Accessed: 16-Oct-2022].
[3] “Discover events happening in your city,” *AllEvents.in*. [Online]. Available: https://allevents.in/. [Accessed: 16-Oct-2022].
[4] *UNATION*. [Online]. Available: https://www.unation.com/. [Accessed: 16-Oct-2022].
[5] “No-fee sports, concert, theater tickets | tickpick.” [Online]. Available: https://www.tickpick.com/. [Accessed: 16-Oct-2022].
[6] “Last-minute sports, Music & Shows Tickets,” *Gametime*. [Online]. Available: https://gametime.co/. [Accessed: 16-Oct-2022].
[7] “Ticketmaster: Buy verified tickets for concerts, sports, theater and events.” [Online]. Available: https://www.ticketmaster.com/. [Accessed: 16-Oct-2022].
[8] “Buy sports, concert and theater tickets on StubHub!,” *stubhub.com*. [Online]. Available: https://www.stubhub.com/. [Accessed: 16-Oct-2022].
[9] “Features,” Whatsapp. [Online]. Available: https://www.whatsapp.com/features. [Accessed: 13-Nov-2022].
[10] “Calendar,” Google Workspace. [Online]. Available: https://workspace.google.com/products/calendar/. [Accessed: 13-Nov-2022].
[11] C. Chen, R. Alfayez, K. Srisopha, B. Boehm, and L. Shi, “Why is it important to measure maintainability and what are the best ways to do it?,” *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, 2017.
[12] I. Heitlager, T. Kuipers and J. Visser, "A Practical Model for Measuring Maintainability," 6th International Conference on the Quality of Information and Communications Technology (QUATIC 2007), 2007, pp. 30-39, doi: 10.1109/QUATIC.2007.8.
[13] B. Thorne, “Four nines and Beyond: A guide to high availability infrastructure,” *Work Life by Atlassian*, 26-Oct-2020. [Online]. Available: https://www.atlassian.com/blog/statuspage/high-availability. [Accessed: 16-Oct-2022].
[14] “Where do websites store passwords?,” *The JavaScript Diaries*, 18-Nov-2019. [Online]. Available: https://www.jsdiaries.com/where-do-websites-store-passwords/. [Accessed: 17-Oct-2022].
[15] C. Song, “Scalable systems 101,” *Educative*. [Online]. Available: https://www.educative.io/blog/scalable-systems-101. [Accessed: 17-Oct-2022].
[16] “Does page load time really affect bounce rate? - pingdom,” *pingdom.com*. [Online]. Available:

https://www.pingdom.com/blog/page-load-time-really-affect-bounce-rate/. [Accessed: 17-Oct-2022].
[17] J. Gaubys, “What percentage of internet traffic is mobile? [Sep '22 UPD],” *Oberlo*. [Online]. Available: https://www.oberlo.com/statistics/mobile-internet-traffic#:~:text=As%20of%20August%202022%2C%2053.74,46.26%20percent%20coming%20from%20desktops. [Accessed: 17-Oct-2022].
[18] “SDLC - Waterfall model,” Tutorials Point. [Online]. Available: https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm. [Accessed: 13-Nov-2022].
[19] “Homepage,” Figma. [Online]. Available: https://www.figma.com/. [Accessed: 13-Nov-2022].
[20] “What Is a Kanban Board and How to Use It? Basics Explained,” Kanbanize. [Online]. Available: https://kanbanize.com/kanban-resources/getting-started/what-is-kanban-board. [Accessed: 13-Nov-2022].
[21] *Spring Framework Documentation*. [Online]. Available: https://docs.spring.io/spring-framework/docs/current/reference/html/. [Accessed: 13-Nov-2022].
[22] “Stack overflow - where developers learn, share, & build careers.” [Online]. Available: https://stackoverflow.com/. [Accessed: 13-Nov-2022].
[23]“Simply easy learning at your fingertips,” *Online Tutorials Library*. [Online]. Available: https://www.tutorialspoint.com/index.htm. [Accessed: 13-Nov-2022].
[24] “Getting started,” *React*. [Online]. Available: https://reactjs.org/docs/getting-started.html. [Accessed: 13-Nov-2022].
[25] *YouTube*. [Online]. Available: https://www.youtube.com/. [Accessed: 13-Nov-2022].
[26] *Google*. [Online]. Available: https://www.google.com/. [Accessed: 13-Nov-2022].
[27] “Google scholar.” [Online]. Available: https://scholar.google.com/. [Accessed: 13-Nov-2022].
[28] *Data Science Stack Exchange*. [Online]. Available: https://datascience.stackexchange.com/. [Accessed: 13-Nov-2022].
[29] D. J. Daly and D. J. Daly, “Economics 2: EC2,” *Amazon*, 1987. [Online]. Available: https://docs.aws.amazon.com/ec2/index.html. [Accessed: 13-Nov-2022].
[30] M. Hollands, “Amplify,” *Amazon*, 2015. [Online]. Available: https://docs.aws.amazon.com/amplify/latest/userguide/to-add-a-custom-domain-managed-by-google-domains.html. [Accessed: 13-Nov-2022].
[31] “Build ci CD pipeline with GitHub Actions and AWS CodeDeploy to deploy node.js app | zero to hero,” *YouTube*, 02-Jul-2022. [Online]. Available: https://www.youtube.com/watch?v=cxTg29ze1D0&ab_channel=Scale-UpSaaS. [Accessed: 13-Nov-2022].