Bilkent University

Department of Computer Engineering

Senior Design Project

Pengout: A mobile app to find events and friends

High-Level Design Report

Damla Eda Bıçakcı
Babanazar Gutlygeldiyev
Sine Mete
Pegah Soltani
Hasan Selim Yağcı

Supervisor: Can Alkan
Jury Members: Varol Akman and İbrahim Körpeoğlu
Innovation Expert: Mehmet Çakır

High-Level Design Report
May 17, 2019

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

1 Introduction

This is the High-Level Design report for Pengout. This report starts with an introduction, in which we provide the purpose of the system, design goals, definitions, and an overview of the Pengout. It is followed by the current software architecture of the applications in the field. Afterward, we provide a detailed architecture of our system. We finalize the report with subsystem services, glossary, and references.

1.1 Purpose of the System

The search for spending free time and getting social is now a more complicated process. Widening cities, crowding places lead to varieties of events taking place around to eliminate according to personal interest and taste. There are several ways that people find the event to go such as asking to a friend, seeing an announcement through social media or using search engines.

Researchers have shown that people have the tendency to use social networking platforms to reach what they are looking for. In a survey, 41% of people respond that they are asking questions in the type of recommendation, invitation, and social connection and topic-wise 36% of respondents ask questions related to entertainment, places, restaurants, and current events. The survey also has shown that the motivation of people to use social networking instead of search engines has three main aspects: trust, subjectivity and social connection [1].

However, the motivation of answering questions is not that high and this results in non-responded questions. Thus, the idea of a social platform that works without the motivation of responding but a motivation of attending to social activity and help other people find their interested events may address such demand.

This search for attending events has another major factor besides finding it, which is to motivate or demotivate an individual to attend a social/cultural event is having or not having to attend it without a companion. In other words, an event is more enjoyable for individuals to attend if they have a friend to accompany them.

There are several portals and applications for people with social anxiety to communicate with each other. However, they are not specifically intended to encourage people to attend events and activities. Given that social activities have more potential to improve our social skills, as well as cultural and educational aspects, a social platform that is specifically designed for those activities, is potentially more effective. So this was our motivation for designing a project as such.

1.2 Design Goals

1.2.1 Usability

- The application will be easy to use and learn within a couple of minutes. Therefore, a user interface must be simple enough. Users will have an option to login with their Facebook or Google accounts with no need to create an account and verify it, otherwise, the application may not be user-friendly.

1.2.2 Reliability

- Failure management: There should be a strategy for error detection and for correction, which begins with the addressed definition of possible errors.

- Up-to-date content: Another specific requirement is providing up-to-date events for recommending to the user.

- Performance: The response time is significant in terms of both usability and reliability requirements. The system should handle a number of requests comes from a number of users within seconds.

1.2.3 Extensibility

- The application will be easy to add features (e.g. the idea of turning tickets to someone, sharing vehicles or flats and etc.). Hence, it should be easy to update with respect to both its background implementation and user interface design.

1.2.4 Privacy

- User information will be private. Unless taking the permission of the user, user information should not be accessed by anyone. Also in the application itself, if users want to hide their information (event history, location, age, interests, and etc.), they should be given this option.

- The security concern for user accounts also aims the privacy of users. Thus, an appropriate login mechanism should be used to avoid such attacks that try to access users account and the system catch and acknowledge the user about the security issue.

1.2.5 Flexibility

- The application will be suitable for users in any city, region or country. It should also provide offline features regarding different user expectations.

1.2.6 Portability and Accessibility

- Our focus group is any people want to socialize and attend events.

- Pengout will be a portable mobile application that works on any Android devices without requiring so many qualities to access a wide range of users.

1.3 Definitions, acronyms, and abbreviations

UI: User Interface.
API: Application Programming Interface
HTTP: HyperText Transfer Protocol
SQL: Structured Query Language
Server: The part of the system for data management and manipulation.
Client: The part of the system for users' interactions.

1.4 Overview

Pengout is an application in which encourages users to attend more events and socialize. Many people would not attend an event because they do not have a companion to go with and they see themselves socially incapable of socializing at the event. Our application focuses more on the social aspect of the events. By using Pengout one can find the events near their locations and be notified about them.

According to the user's interests (which will be specified by the user in the very beginning of creating their accounts) and event history, the system will also suggest events. After the event is marked by the user the system will find a companion who is attending the same event. So both users will be able to communicate through the app before they attend the event. This will not only be relaxing for the ones who are not comfortable with socializing in real life, but it will also encourage many people to get to know new people while attending events.

In this application, the users will be able to create their own events and set it public, in which all the other users will be able to view or set it private in which the users will be able to share with their own friends.

2 Current Software Architecture

There are many applications in the social network market to help people for their social life. In this section, we examined some of them, which are similar to our social network app.

- MeetUp: MeetUp is a web-based application to socialize with people according to your interests. It has many social groups like book clubs to nature walking groups that you can participate or you can create your own group and meet up with new friends who share your interests [2].

- Circle**:** Circle is a mobile application, which is a location-based social network. It uses GPS to find your location and you can easily meet new people in your general area [3].

- Socialradar**:** SocialRadar is a location-based application that gives notifications about who is around you when you go to an event or any social places (restaurant, bar, club, etc.). This app can access your social media contacts and gives some helpful reminders of their details [4].

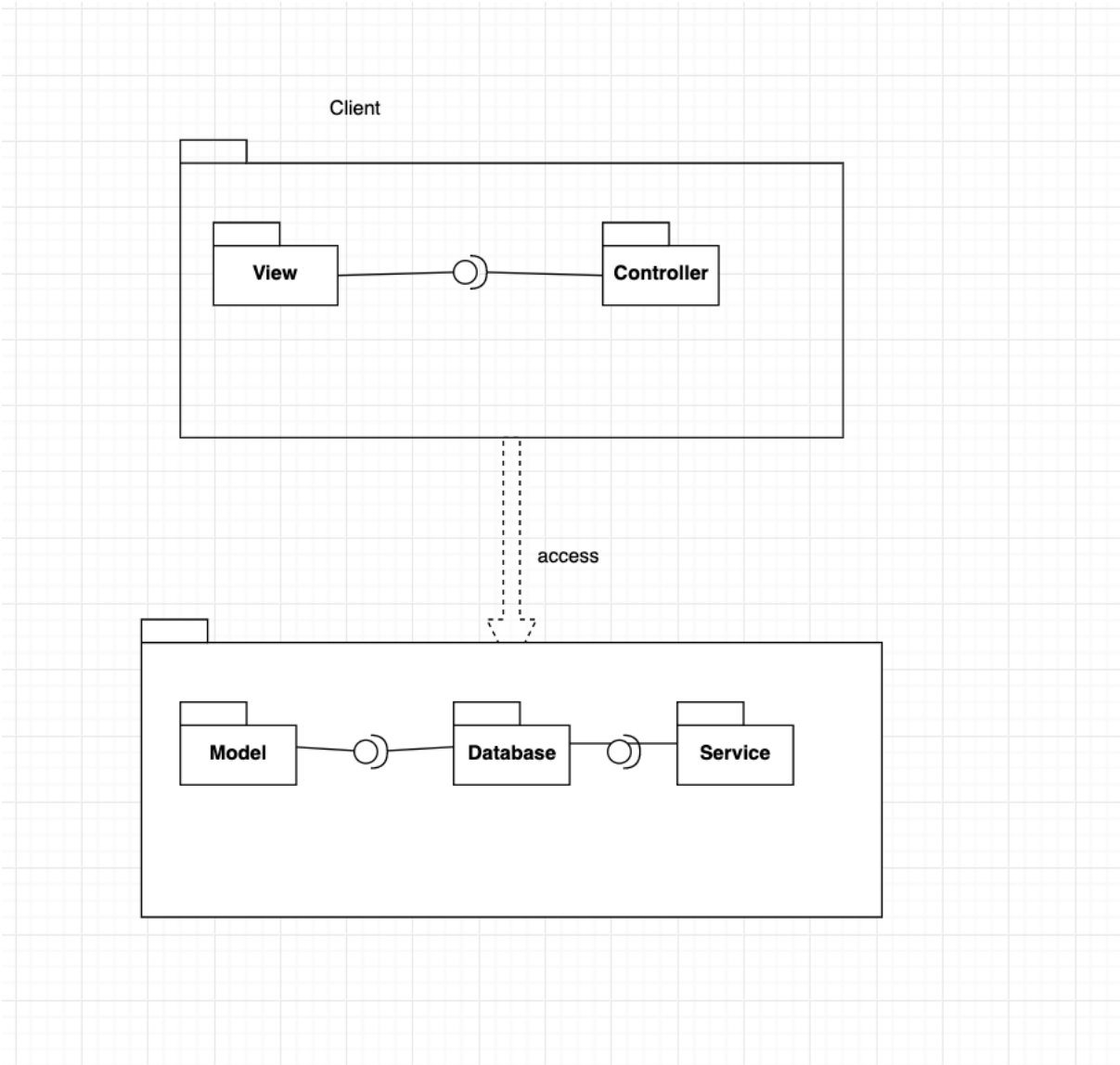
3 Proposed software architecture

3.1 Overview

This section of the report contains detailed information about pengout's system. We will begin with subsystem decomposition and hardware/software mapping. Secondly, detailed information about persistent data management and access control will follow it. Finally, security, global software control, and boundary conditions sequentially.

3.2 Subsystem decomposition

According to our specifications, we decided to implement Pengout using client/server architecture. We want to apply client/server architecture because Pengout needs to respond in a reasonable time (e. g. delay must be at most 1 second) to a large

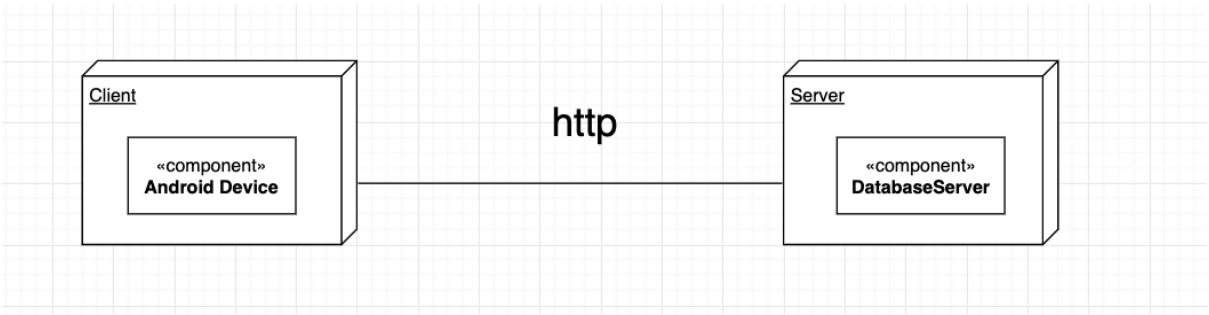


number of users. We can accomplish this by separating the system into two parts, namely client and server. Thus, we can optimally separate functions that are relevant to the client and the ones that are relevant to the server only. For example, Pengout has recommendation algorithm implemented to suggest personalized events. We can run this algorithm in the server and give the client only related results. When we do these, the client should not experience any delay.

Furthermore, we will apply the Model-View-Controller (MVC) design pattern to systematically develop Pengout. It will help us to deploy to the client only sufficient and necessary components of Pengout while providing a fully functional application. Precisely, we will place only views and controllers to the client side. We will provide other services on the server side with the help of communication between controllers and the model subsystem. The server will contain Model, Database, and Services subsystems. The model subsystem will provide communication between controllers of the client side and a database of the server side. The database subsystem will hold data about users and events of the Pengout. Moreover, the results created by the services subsystem will be stored in the database. Additionally, we have the services

subsystem to implement service functionality of Pengout separate from others. Services subsystem has a recommendation system, Facebook API, and Google API parts in it. Events, friend list, etc. will be fetched from Facebook and stored in the database to utilize for the other parts of the system. For example, the recommendation system will create some information about some users and events. It can be quantitative percentages whether to recommend a specific event to a specific user. These results will also be stored in the database to be queried later by model subsystem.

3.3 Hardware/software mapping



We have two main hardware side: client and server. Client-side communicates with end-users via Pengout UI. Moreover, all data management and manipulation operations are done in the server-side which is hosting the application. Client-side needs to communicate with the server to get the required data so this communication will be provided by HTTP technology.

3.4 Persistent data management

We have large data that needs to be managed and manipulated. We hold information about the user's name, city, friends list, and past events. Also, we need accurate information about users' interests for our recommendation and match features. Therefore, database management is a very important issue for this project and we solve this problem, we will use MySQL by generating a relational database. Therefore, we can manage and manipulate the database efficiently.

3.5 Access control and security

Pengout should be secure enough to work correctly and efficiently. Therefore, security and access control are very important issues. Our system does not require too much personal information. It requires just users' email addresses, passwords, name, and age information, etc. Password information will be encrypted by using some hashing mechanisms. Also, each user is responsible for just their own actions, means users cannot manipulate the other data, which is not accessible. And that information is kept in the database which is just accessible

by server side. The client side is not allowed to reach data, the client side is just responsible to communicate with the server side. And all database operations such as manipulation, management are done by server side. Therefore, our system is very secure against end-users failures.

3.6 Global software control

For Pengout, as global software control, the centralized real-time control system will be used. Our system is not a soft real-time system but there will be two types of recommendation, so two types of handling big data, one for matching, one for event recommendation. When a new event is created, our system must evaluate it immediately. However, in specific time intervals, the recommended events to the user must be reevaluated; those processes will be started or stopped according to system state variables. The system controller process decides when processes should be started or stopped depending on system state variables. Those system variables controlled by the user input and our preference of time interval. When the user logins to the system, the inputs of the user checked by the Database in the server.

3.7 Boundary conditions

3.7.1 Initialization

The users must download our application on their phone running Android operating system in order to use it. To start the application, the user must create an account via Facebook, Google account or email and use that account to log in to the application. Users cannot use the application if login fails. Additionally, the application requires an internet connection in order to control registration information, taking initial real-time event data, saving user profile information.

3.7.2 Termination

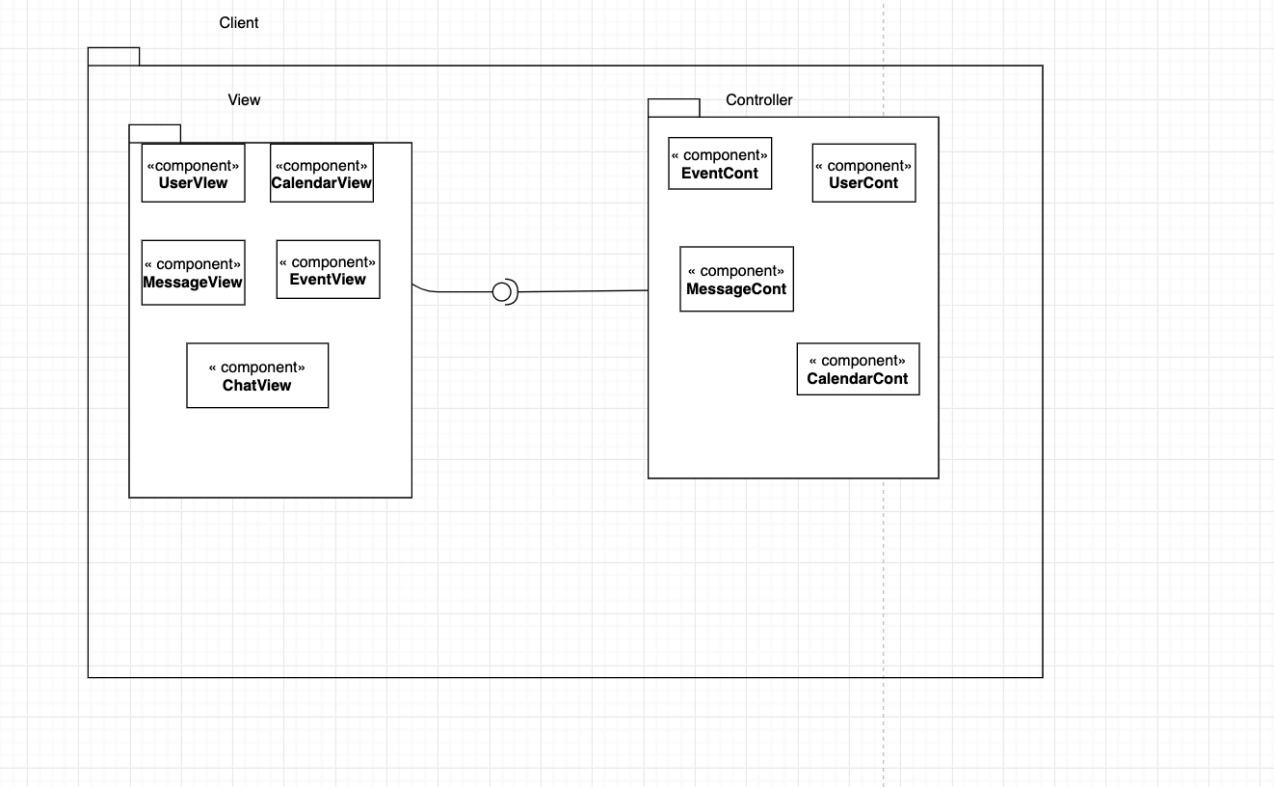
Users can terminate the application by logging off. If the user wants to keep logged in, the user can choose not to log out, then the user can terminate the application by exit pop-up window or the user terminates the session by clearing the application data. If the user does not close the application, it will continue to run on the background.

3.7.3 Failure

The application fails if there is no internet connection since profile information and real-time data cannot be obtained.

4 Subsystem Services

4.1 Client



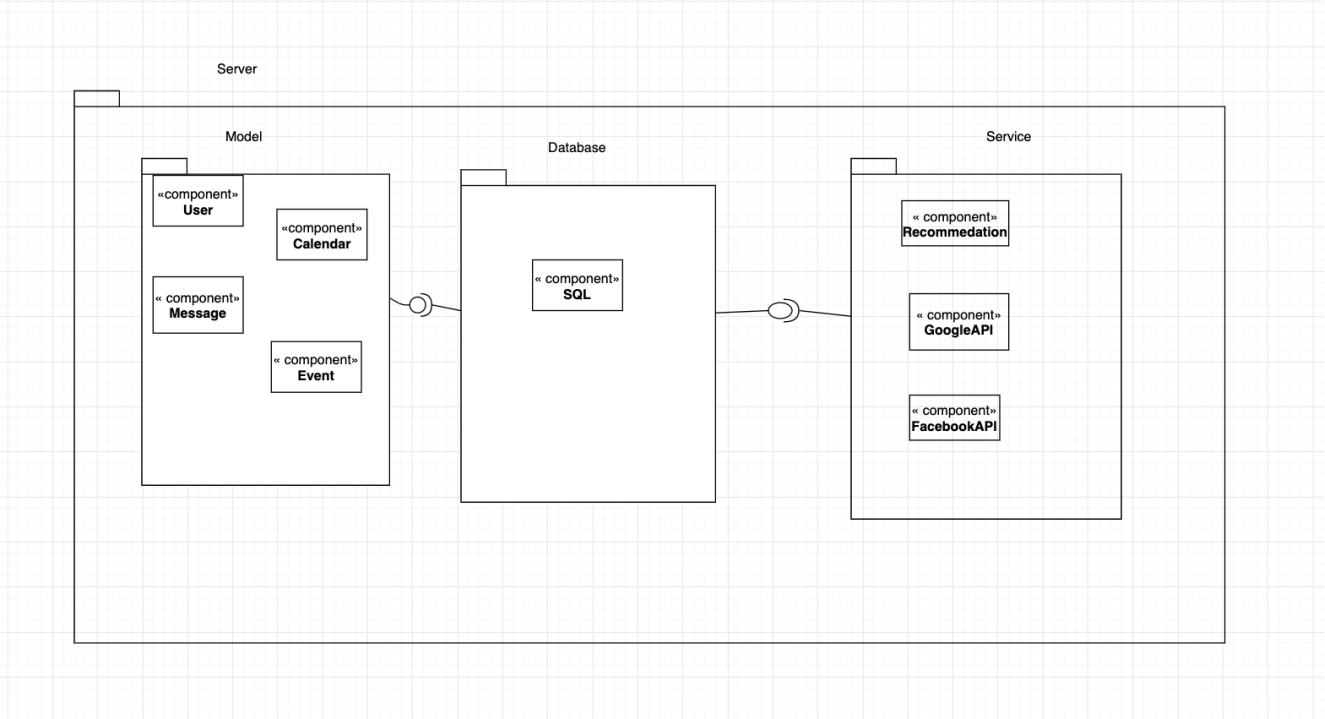
4.1.1 View

UserView: It displays and controls the user view.
CalendarView: It displays and controls the calendar screen..
MessageView: It displays and controls the messages screen.
EventView: It displays and controls each event's view.
ChatView: It displays and controls the chat view.

4.1.2 Controller

EventController: This class manages interactions from the view related to event data and control the responses to the server and from the server back.
MessageController: This class is responsible of managing the message interaction and updates both view and model according to the interaction from frontend.
UserController: This class updates user accounts and related model components according to the request comes from the view.
CalendarController: This class is in charge of interaction between calendar view and related components in server.

4.2 Server



4.2.1 Model

Event: This class keeps information and sets the functionality related to events.
Message: Message objects keeps date, message body,sender and receiver information.
User: This class keeps user information and the functionality related to users. It contains information including user id, username, password, email, gender, birthday, city, profile picture, activity list, friends, calendar id, location, interest list.
Calendar: This class holds the information of dates, user id, list of events added to the calendar, and related information.

4.2.2 Database

This subsystem keeps the data about the users and events of Pengout. Whenever there is a new user or a new event it is verified by the Database and stored in it. Moreover, the Database subsystem will respond to queries of the Model and Services subsystems. The model subsystem will make queries to respond to the Controllers of the client side, while Services subsystem will query to optimize recommendation system and to save user data that is fetched by Facebook API and Google API.

4.2.3 Service

Recommendation: This class keeps track of the related updates in data regarding user activity, event information, friend list, and user information, and manages related responses in the backend side.
Google API: For the signup process, this class implements the connection to Google API.
Facebook API: For the signup process, this class implements the connection to Facebook API.

References

[1] Morris, M.R., Teevan, J. and Panovich, K., What Do People Ask Their Social Networks, and Why?. [online] Available at https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/chi10-social.pdf [Accessed 24 Feb. 2019]

[2] Meetup.com. (2019). About - Meetup. [online] Available at: https://www.meetup.com/about/ [Accessed 24 Feb. 2019].

[3] Circle-local-social-network.en.softonic.com. (2019). Circle - Local Social Network for Android - Download. [online] Available at: https://circle-local-social-network.en.softonic.com/android [Accessed 24 Feb. 2019].

[4] Socialradar.com. (2019). Welcome to socialradar.com. [online] Available at: http://www.socialradar.com/ [Accessed 24 Feb. 2019].