SOCIAL TRENDS

Software Design Description V2.0

MIDDLE EAST TECHNICAL UNIVERSITY

DEPARTMENT OF COMPUTER ENGINEERING

Anil BEKTAS 1824010
Tekin ERTEKIN 1745959
Gozde GOKMEN 1678960
Bashir KAZIMI 1824523

Table of Contents

1. Introduction

1.1 Scope

[1] [2] This Software Design Description (SDD) describes the detailed structure of the components of this social media project and the precise implementation details required to satisfy the requirements as specified in the Software Requirements Specification (SRS). It is assumed that the reader has read the SRS, since this document also defines the implementation details of the desired behavior given the requirements within it. Here is the information about the main parts of our project:

- Software part of the project will consist of three main parts: database, main program and user-interface.
- For now, we do not need any other software program except for the mobile part of the project.
- User-interface: Connection between the users and the system will be provided by user-interface. Besides, there can be small user-interface applications for system managers. JQuery will be used for that.

Main program: This is the part where system provides necessary information for user-interface with the database part. Also, there will be some methods for writing the information that comes from user-interface to database. For the sake of the system in the security manner, no database connections will be established in this part.

Necessary information from the methods which are inside the database part (Node.js).

Database: In this part, the information that is received from main program will be saved and necessary information will be gotten. Data interchanging can only be done from this part. In the other two parts, there will no interchanging data process.

System requirements will be changed according to number of the current users.

1.2 Purpose

The design description defined in this document serves multiple purposes:

- To generate a document about what we are going to do in this project.
- To describe the functional structure, data and algorithms to be implemented.
- To be used to assess the impact of requirement changes.
- To assist maintenance activities.

1.3 Intended Audience

Our intended audience is as follows:

- Provide a software design description document to members of the project responsible for the software development (the developers) of the framework to help guide the development effort throughout the project.
- All students and teachers who are interested in or project.

2. Definitions

USER	A person who can register to system directly without any location constraints. There can be 2 types of account holders: • Personal Account: Targets daily use, enables following friends, getting followed by others • Organizational Account: Initiated by famous persons, companies... Only property belongs to this type of account is getting new followers. The administrators of this account don't have rights to add friends or follow others like what happens in daily usage.
TRIAL ACCOUNT	Temporal situation where anybody can sign in without registering to system beforehand. When the trial user signs out all logs are removed instantly. It is meant for whom are interested in our system and giving a try on line.
TRENDS	Like how Twitter filters the twits while rendering the worldwide trends (or local) we are planning to filter and render the trends according the data within except it doesn't only contains text information but also videos (by now only unique id linked to web link) and images. Each 3 kind of trends are refreshed per 30 minutes. The more we improve the interface and back end us might consider news trends.
HASH-TAG TEXTS	While rendering the text trends we will be only considering the texts starting with hash-tag and

	ending with a space. First new 10 hash-tag text trends are shared on trends panel.
IMAGE	The most shared first 10 images are placed into the image trends. The comparison is on pixel-based. The criterion is not only share statistics but also like and dislike counts.
VIDEO	The same with images but we aren't serving a videos service yet so the only criterion is embedded link to regarding You-tube address.
TRENDS PRESENTATION	The trends are presented to user with statistics within. (like dislike or like count)
POST	Users are able to share the 3 regarding types of contents at their wall with their circles. The posts may include the several types at the same type. We are going to set up this property using So-cl as reference point. A user can share this post via its account without any dependency or make comments to other posts.
PUBLIC COMMENTARY	Say our object might be either text with hash-tag, video or image. When an object becomes eligible to be placed in trends panel users can comment under these trends, either like or dislike. This kind of object are independently published.
CASUAL GROUPS	Users create and gather under a group to make the communication easier. This groups contain posts. There can be two types: Private or Public.
AUTOMATIC GROUPS	School, class, university, village... any kind of community where a user has been included is formed as a group. For instance when a user register to a group as pioneer user when s/he sets its home town to İzmir, a İzmir group is created and attained as group administrator. Whoever changes its home town to İzmir is automatically added to this group. These users can abandon group instantly. Automatic groups are segregated from casual groups while they are displayed so it worth more than a group with respect to users. Sample groups are: METU, Ankara University Administration Faculty Class of 2012 etc.
AUTOMATIC GROUP DOMAINS	City County High School (Year) University Department (Year)

MOVABLE PANELS	Drag and drop enabled panels included in our system canonically. Also resizing is enabled. Panel Parts: 1. Post • All posts • Group posts • Private post 2. Trends • Text • Image • Video 3. Message Panel 4. Suggestion Panel 5. Notification Panel 6. Group Panel
MESSAGE	Like other social networks users can chat. There will be rich content message sending option in the future.
INSTANT CHAT	There will a platform created where people can chat instantly.
DESIGN	Users are able to save the designs which they are formed through dragging and dropping the panels. They can add background to this designs and share with their circles.

3. Conceptual Model

This part is separated to basic concepts and context of the SDD. Conceptual model easies the process of the audience of this document to understand our project. “Electrons orbiting the nucleus of an atom the way that planets orbit the sun”, how the previous sentences models the orbiting of planets within an example it is not necessarily to be correct or precise like in a scientific article manner.

The closest social media conceptually should be Facebook. Let’s say a user has privilege to do what s/he can do at Facebook mostly, commenting, sending likes, sharing posts but with an essential difference: There is also columns where s/he is able to see filtered data from our servers which has the most count, like or even dislike.

There are minor differences like for instance there has trial account access which instantly give anybody to reach our servers and test out the site like a signed up, official user. This kind of enhancement is hard to conceptualize since they are mostly brand new terms at the sector, at least not implied by major social network vendors.

3.1 Software Design in Context

3.1.1 Application Overview

It is a comprehensive social platform host many type of trends and publishes aesthetically to the world. This easies processing the large amounts of information over all Internet backing up mostly in Twitter, Facebook, YouTube and other social platforms. Our system is planned to be one of the most essential filter to this ridiculously rich environment which human brain couldn't ever handle anymore.

3.1.2 Technologies Used

This project will be implemented using mostly Linux command line, Node.js, JavaScript as both server and client side coding like JQuery. Other web programming canonicals are also well-formed through our project. (CSS/HTML/JS). Thanks to Node.js and its rich package management systems for handling all connections, sockets etc.

Also Node.js has a default package registry which take way much usage of. The major packages we use are:

NPM REGISTRY	
MySQL	Handles the all interaction in one spot
Fs-extra	File operation but contains much more than “fs” like more neat mkdirp
Hotnode	Reloads the code automatically
Express	The golden yet efficient package which wraps all server interaction at one hand and does very much chores instead of us like handling routes and requests and other stuff
Stochasm	Random number generator
Postmark	Send the confirmation e-mails for newly signed up users
Socket.io	Handles sockets catching or emitting the data
Formidable	Parses data for file uploads, MIME type is checked
Jade	Converts the html to the language which Express can interpret, html template

3.2 Software Design Descriptions within the Life Cycle

3.2.1 Influences on SDD Preparations

The previous documents has been committed like SRS is used as reference point to this and all future studies.

3.2.2 Influences on Software Cycle Products

The most convenient and urgent revisions are based on requirements of our future users. It is a fact both SRS and SDD may go through small or big revisions however the ultimate goal still stays the same. SDD will influence test documentation (if we need any) and our system.

3.2.3 Design Verifications and Design Role in Validation

Design should be verified and validated to ensure the system fulfills specified requirements and serves for planned use. While we are constantly iterating with testing our code at local host in the future if it is found convenient test scripts should be implemented to test pretty complicated URL net and buttons that have rights to change underlying databases.

4. Design description Information Content

4.1 Introduction

This document is prepared for the design of “Social Trends”. The document holds the Identification, diagrams, user views and user viewpoints.

4.2 SDD Identification

“Social Trends” is a social network that provides people with a better way of communicating. Social Trends has all the basic features that other social networks do; like instant messaging, sharing posts, photos, commenting, text trending, multiple posts, liking posts, etc. In addition to all these, Social Trends has some features that are specific to itself only. These are trial account, movable panels, image trending, video trending and many more.

4.3 Design stakeholders and their concerns

Developers, testers and probably the end users are stakeholders of Social Trends.

4.4 Design views

For modeling the diagrams for views, UML is used. It stands for Unified Modeling Language. UML is a general-purpose modeling language in software engineering. It uses graphic notation techniques to create visual models for software systems. More information about UML could be found here: http://tutorialspoint.com/uml/

4.5 Design viewpoints

The context, logical, state dynamics, interaction, and interface viewpoints for Social Trends are described in this document.

5. Viewpoints

5.1 Context Viewpoint

Context viewpoint shows the functions provided by the system. System boundary and the interaction between the system and users are also shown through this viewpoint. The context is defined through the interactions between the end users and the system. At this viewpoint the actors shouldn't know the inside of black box but only their limits. There are several figures depicts in order personal actions, interaction with each other, more miscellaneous properties and system administrators use cases.

The most anticipated subjects and use cases are depicted below. Actors are completely ignorant of the underlying functions of the buttons that make them take an actions until we declare an Open API in far future.

The output and input is generally text based including video and text trends itself since video trends are just a link. Only difference is noticed at picture uploading and sharing a post containing a picture medium. The MIME types will be handled accordingly by the server.

At the picture depicted below a user is able to share images, videos and texts with his or her friends. Login, sign up or even trial usage is allowed to anyone willing to see our site.

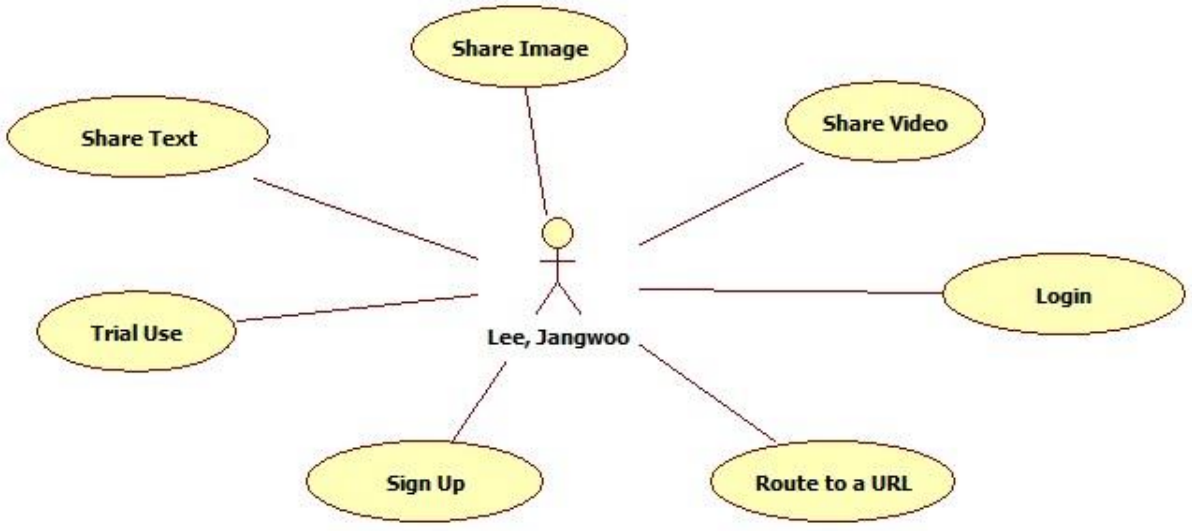


Figure 1 Personal Actions

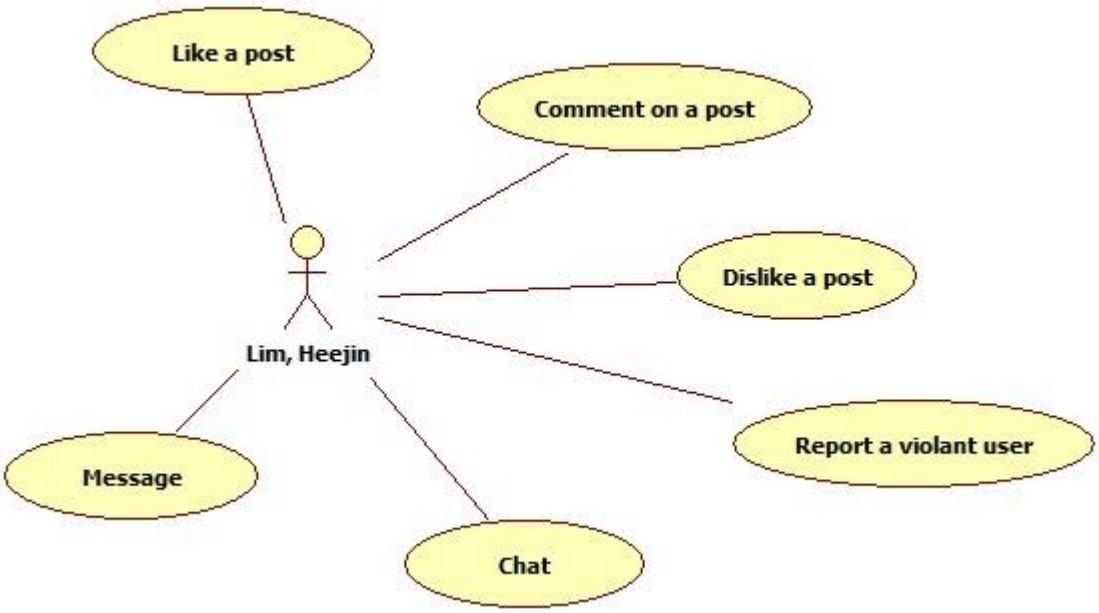


Figure 2 Interaction with Others

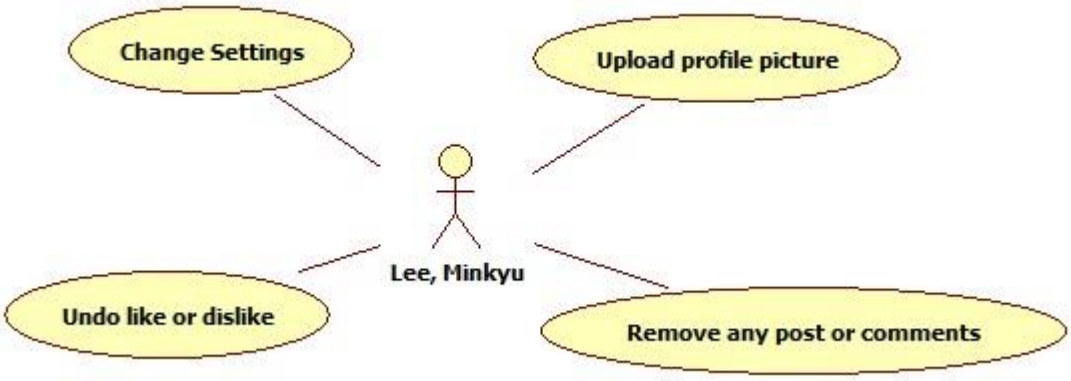


Figure 3 Miscellaneous Actions

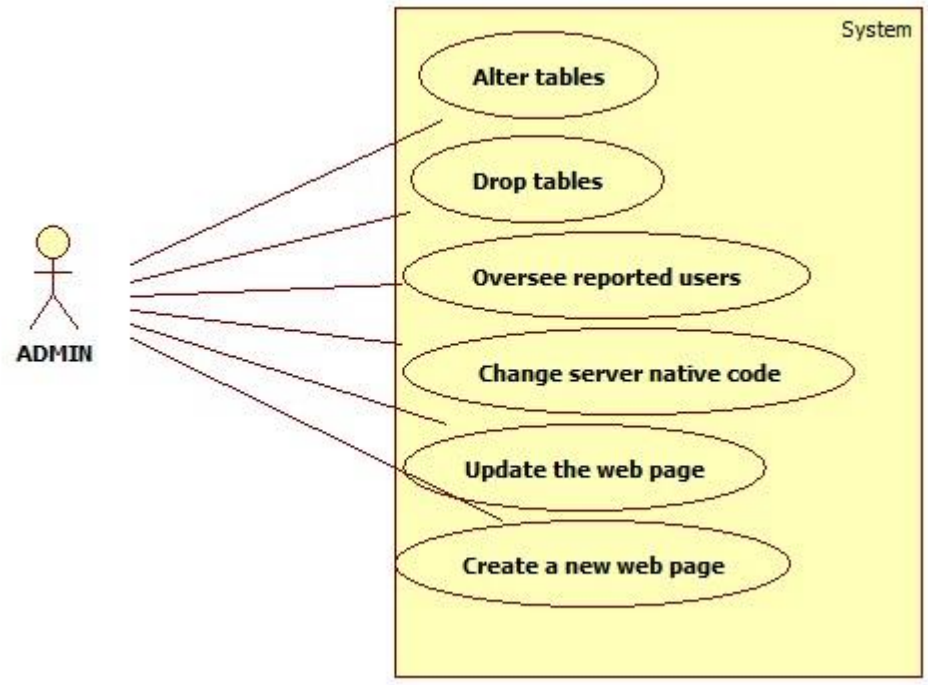


Figure 4 Admin Use Cases

5.2 Interaction Viewpoint

The main flow is displayed elegantly under, it is required to put this diagram into words to be more precise. It has been a cliché that you must register to be part of a system but in our system this not the only way. We are introducing “trial account” property to make users experience in a closer look to our system. Since we conclude that we patched the gap that is not shown below let try to interpret the table. The home page, other than trial account serves after you sign up and at home page where the action starts like from how the trends are displayed in a trim, fluent way to interacting with your friends.

The user page serves for personal settings and also there may be posts belong to this person (i.e. like how “only me” property of Facebook works)

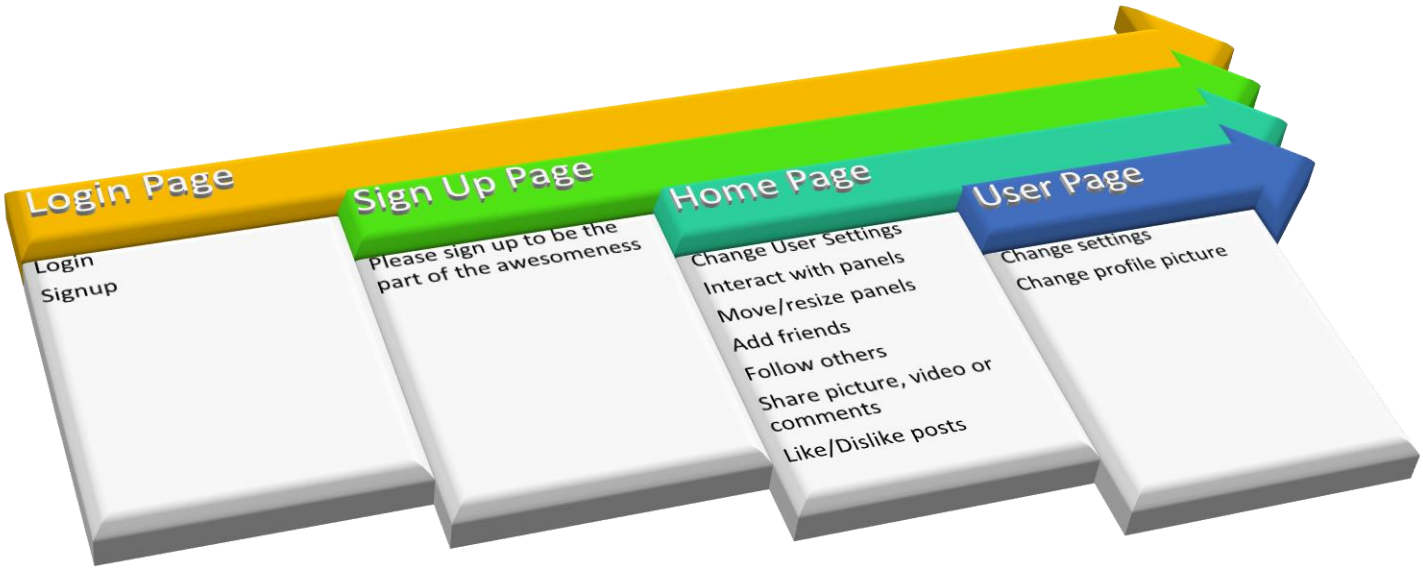


Figure 5 Interaction Flow Chart w/o Trial Account Property

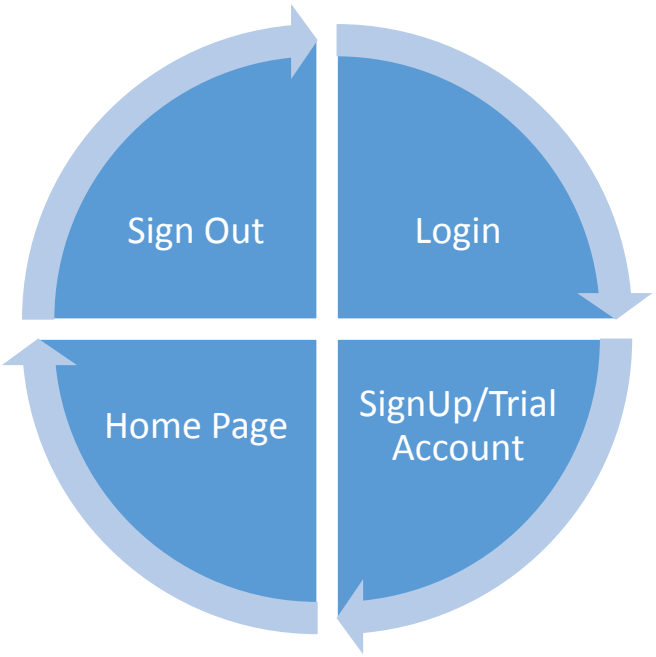


Figure 6 Canonical Cycle

5.2.1 Design Elements and Constraints

The other part should be mentioned here is concurrency and synchronization of the posts and the underlying comment/like mechanism. The server should handle like how multiprocessing is handled by any operating system, treating the users as a process and blocking/allowing some operations. As IEEE standards recommends we are including this topic.

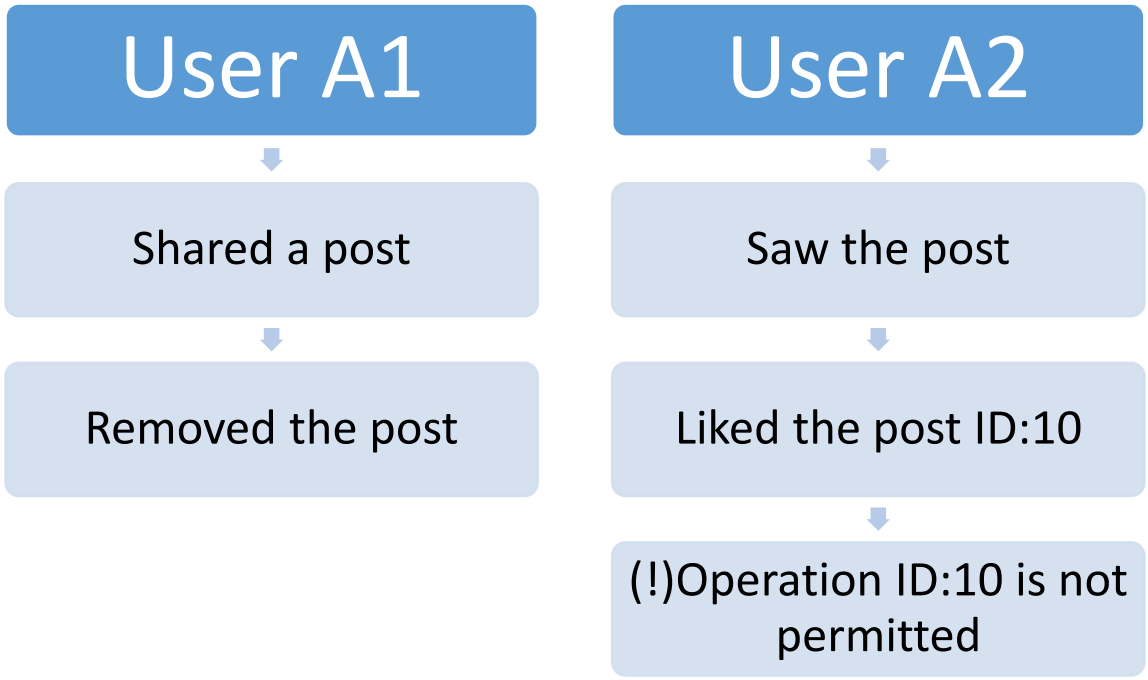


Figure 7 Concurrency Example 1

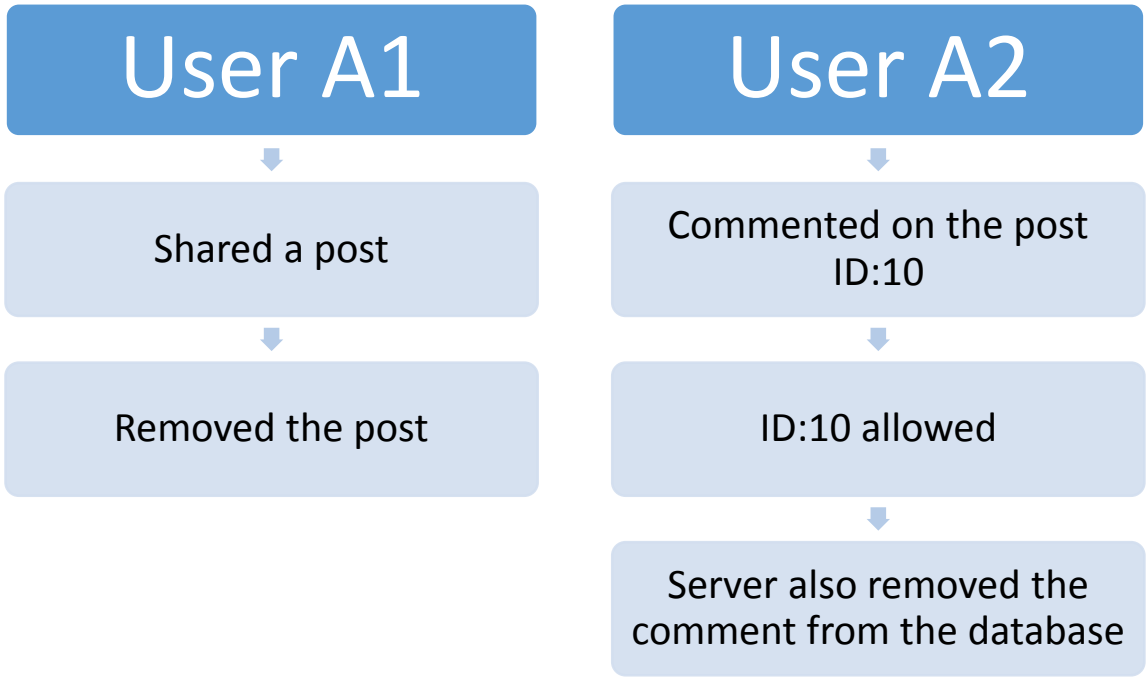


Figure 8 Concurrency Example 2

After concurrency the most challenging hassle is supposed to be filtering effectively the trends backed up in our server to display qualified results. The flow of the data is as follows at initial stage there are free moving data but (note that there isn't one output but 3, the diagram is only for symbolic purposes) filtered afterwards.

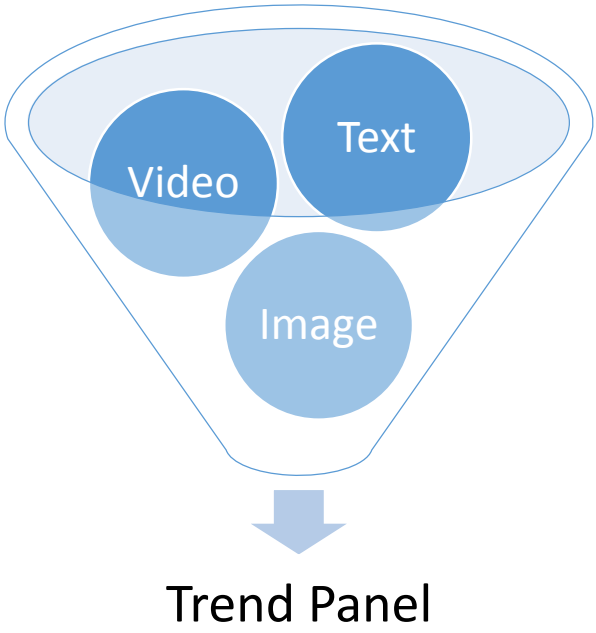


Figure 9 Data Flow of Filtered Trends

5.3 State dynamics viewpoint

In order to explain what is going on inside application, we draw some state transition diagrams to show relationship between interface, database and main program. We show adding friends, commenting, liking and disliking post and comments in these diagrams.

5.3.1 Adding posts

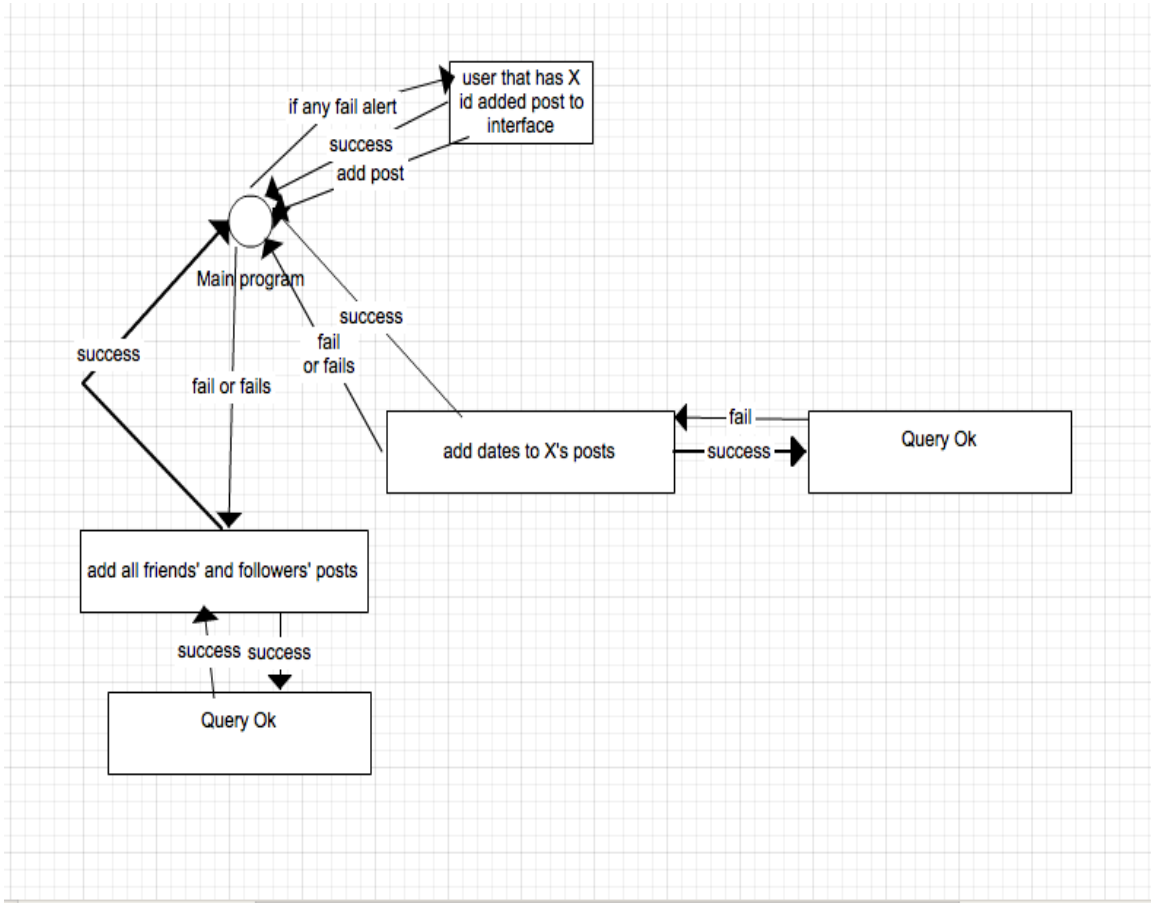


Figure 10 Pipeline of a Post

When there is a post, main program will add post and its date to database where a user that has an id X has a record and put this post under X' s post list. If there is a fail in database it will return to post state. Also main program will add posts to all friends' and followers' posts lists when successfully returned from database state.

5.3.2 Adding Friends

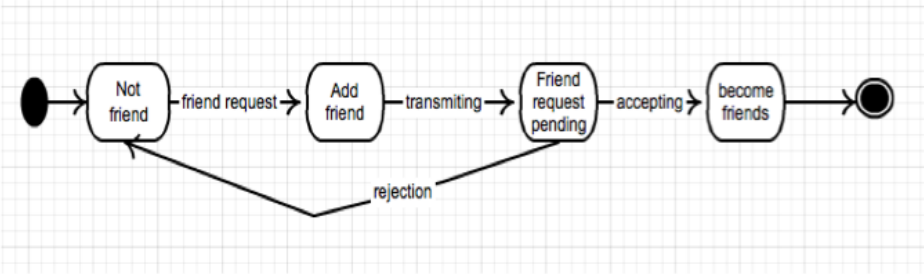


Figure 11 Adding Friend Pipeline

A user X that has an id adds the other user Y that has a different id. User- interface part transmits this request to main program. Main program make database to save this request. And database save it in a list. Later the user that has a friendship request sees it and confirms it. User-interface sends information that Y approved X's request to main program. Main program makes database part to remove this request from the Y' request list. Also, main program makes database to add Y to X's friend list. In the same way, main program makes database to add X to Y' friend list. And later on, database part does processes about data interchanging.

5.3.3 Comments

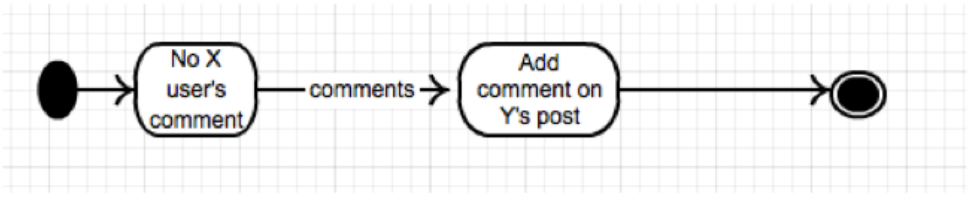


Figure 12 Comment Pipeline

A user X comments on user Y's post that has an id Z. . . User-interface part transmits this post to main program. Main program make database to save this post. And database save it in a list. There is no need to have an approval this time. Also, main program makes database to add X' comment to Y's Z post. At the end, this post will be added to database.

5.3.4 Comments Like or Dislike

When there is a comment main program will add comment and its date to database where a user that has an id YYY has a record and put this comment under YYY' s comment list. If there is a fail in database it will return to comment state. Also main program will add comments to comments list of all friends' and followers' posts lists when successfully returned from database state.

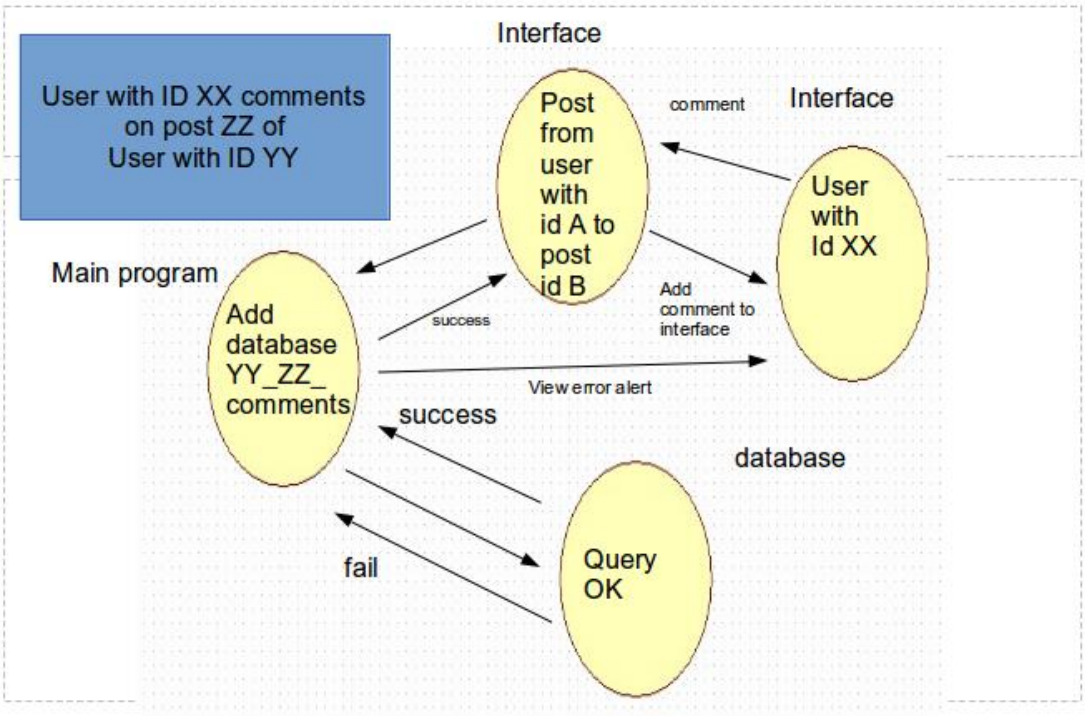


Figure 13 Like and Dislikes under any Comment

5.4 Interface Viewpoint

As you can see, from the MVC login, users are setting input through an interface. Business logic is running on server, view is running on interface and controller is where a user is able to edit the corresponding fields in database with limited permissions. Admin has direct access to database. The following diagram show the general structure and after this the document will gradually reveal the valid

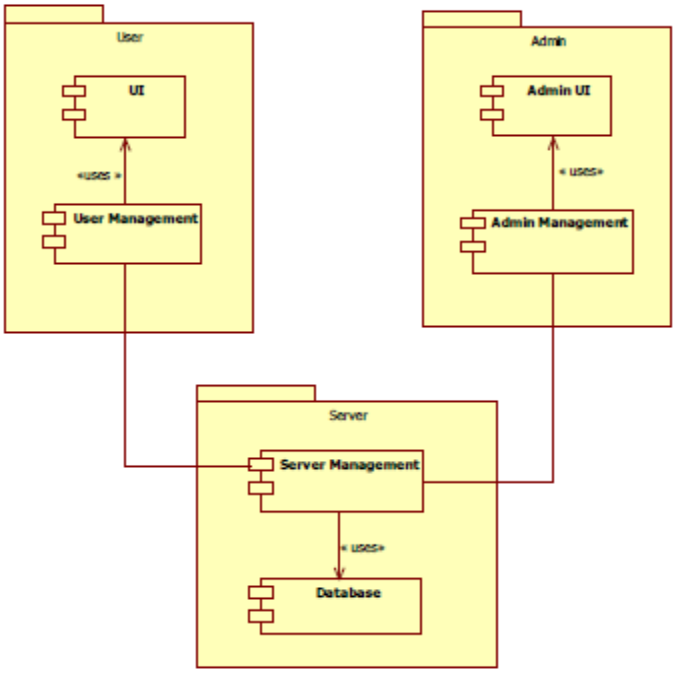


Figure 14 MVC Structure

…input and outputs and also scenarios and how a consumer perceives the system. The interface viewpoint in summary is meant to display behaviors of the interface and its sub components to ease the work planning task between testers, coders and designers.

5.4.1. User Interfaces

The interface below are yet premature but basis instances of the sign in and sign up pages. We are planning to donate with nice, beautiful-looking Twitter Bootstrap framework to these pages. The most important point under this title is background. With help of the background we are aiming to reflect the atmosphere of socially appealing places on our system. Under we demonstrate some shots from our first page. You can see how this page evolved and it will be evolving likely at this path so on so forth.

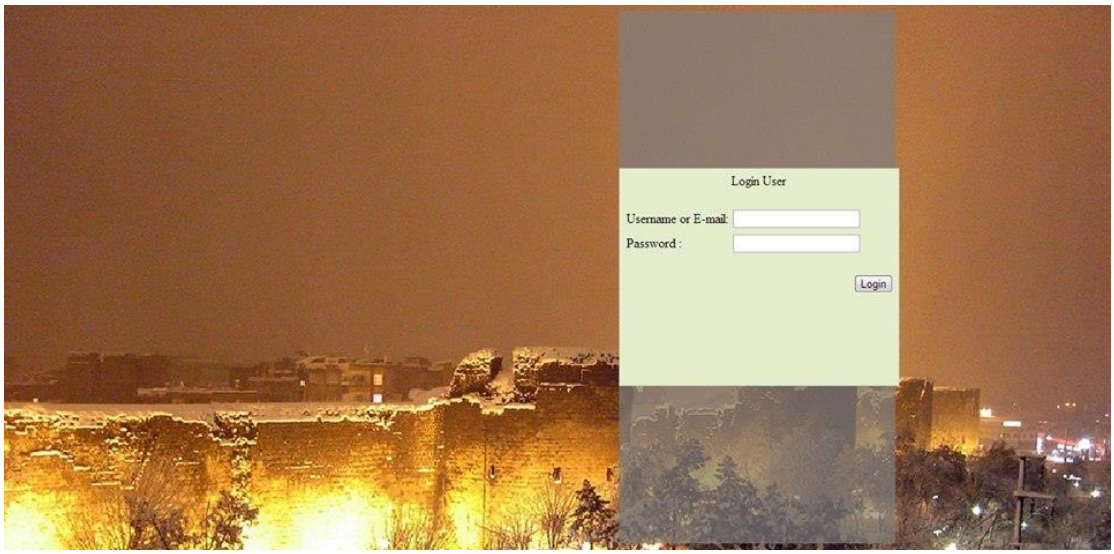


Figure 15 Version with Ankara Castle at Night

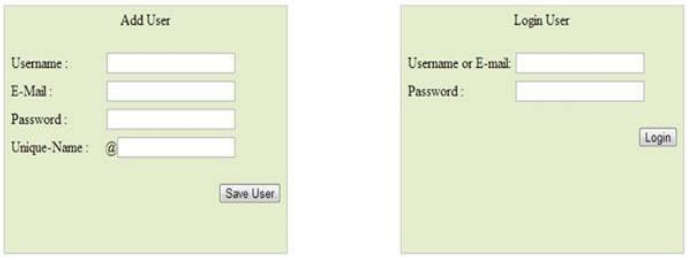


Figure 16 a very Raw Sign up Page

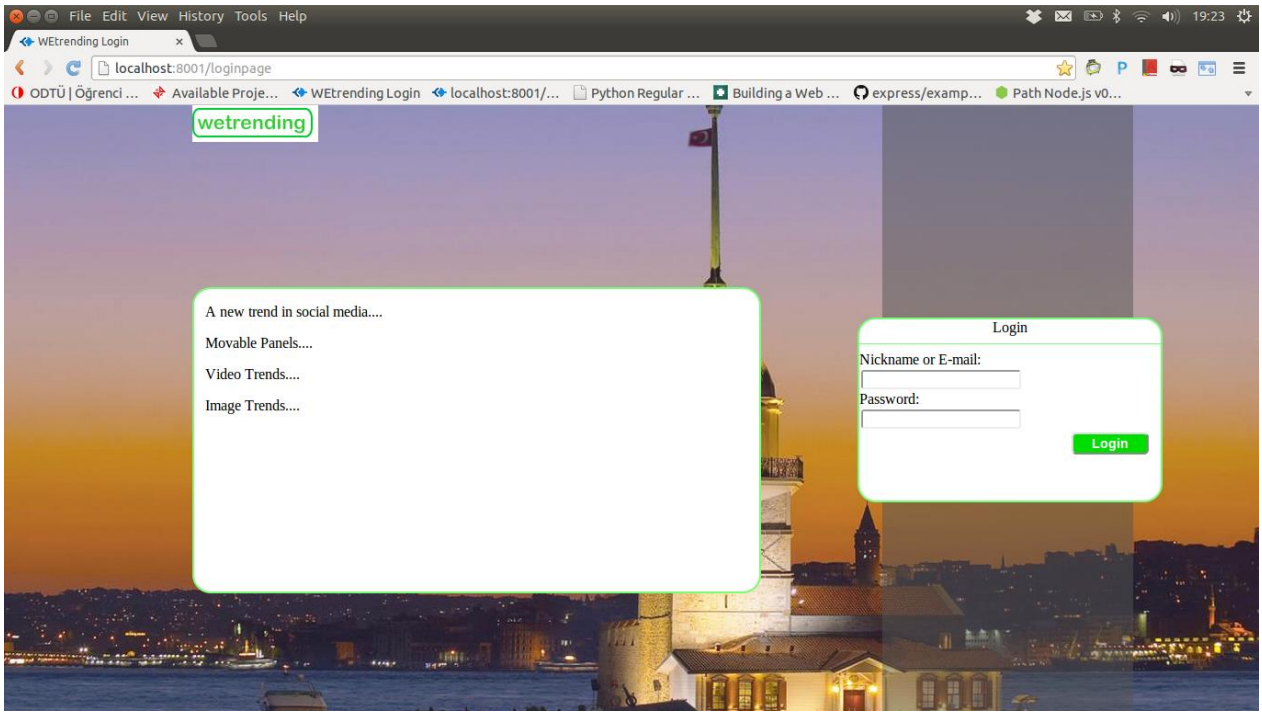


Figure 17 Login Page with Maiden Tower Background

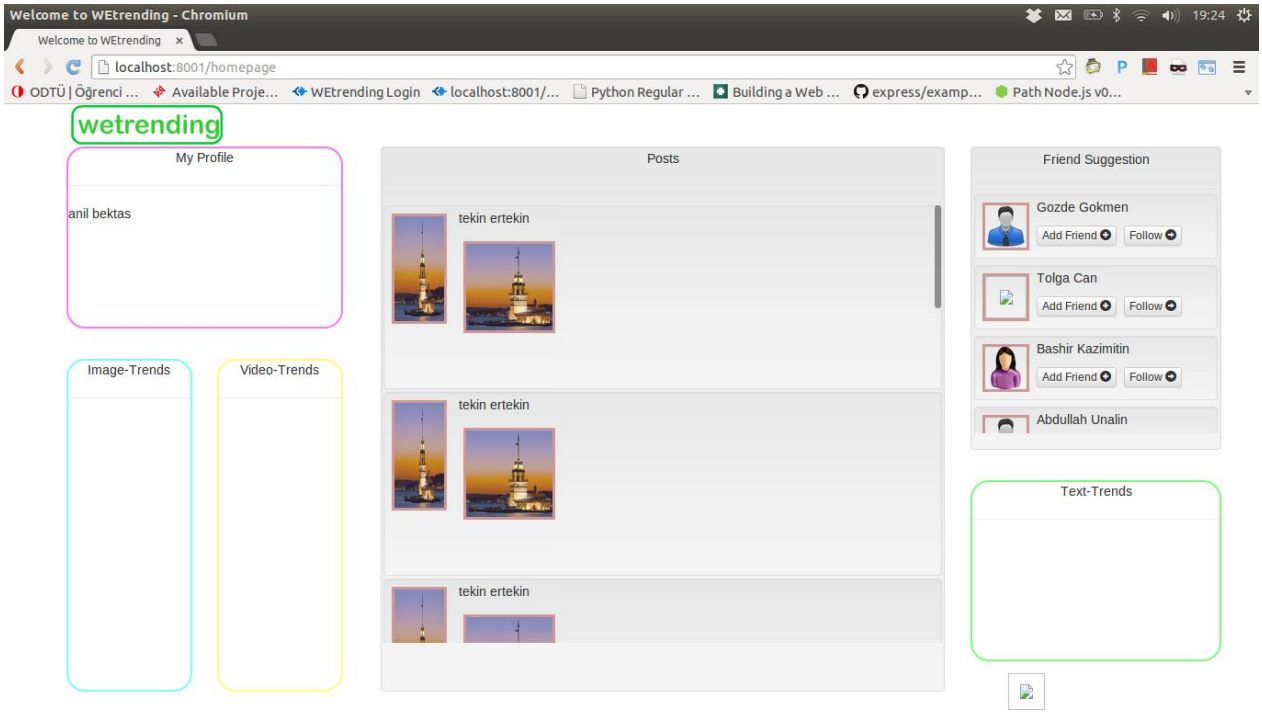


Figure 18 Home Page

5.4.2. Valid Inputs and Outputs

Below how each HTML component reacts to an interaction is enlisted. The table names refers to at what page the users make the interaction enlisted at the same table and at right the output of the action is displayed. As a rule of thumb interface viewpoint show this scenarios related to interfaces and data types.

At login page a user…

enters wrong nick	Specific Alert
enters wrong e-mail	Specific Alert
enters wrong password	Specific Alert
enters correct nick and password	Nick processing calls are made and navigated to the homepage
enters correct e-mail and password	E-mail is processed for global standards and the user is navigated to the home page
clicks "register here" etc.	Navigates to registration panel
clicks the login button	Processes the submit fields.

At sign up page a user…

enters valid email and fills other areas	Server sends a confirmation e-mail
enters wrong email and fill other areas	Server doesn't send a confirmation e-mail since it is able check to improve effectiveness of itself

At home page a user…

clicks his name	Navigates to his or her user page
scrolls down in the friend suggestion field	New friends are suggested
clicks sign out	Cookie information is removed and navigates to sign in page
sees trends panels	Text, video and image trends are enlisted
sees posts panel	The posts are enlisted
drags the panels	Panels are interchangeable and resizable
clicks on a post	Post sub window is popped out for adding comments
clicks on post like/dislike	Like and dislike counts are changed of the related posts
scrolls down the trends panel	New posts are enlisted

At user page a user…

clicks on change profile picture	Upload dialog box opens.

clicks (or fills) password fields	Changes the user's password code
clicks to change privacy settings	Server records the privacy related changes like which audience a user is intended to share his or her posts or later on 2 step verification might be enabled
clicks (or fills) name and surname fields	Sets his or her name and last name
clicks home page link	Turns back to the homepage

5.4.3. Data Attributes

These are the set of attributes which belongs the input fields mostly (say the part a user interacts) and the table gives brief introduction to how these data types are perceived by the system. This kind of text data generally is recorded via a cookie to diminish the burden of the server. The data flow of the databases are not included.

Name	Range	Type	Error Code	Input	Output
email	{2-6}@*	String	*Not an email Not registered*	YES	
name	{100}	String		YES	
surname	{100}	String		YES	
nick		String	*Not registered*	YES	
password	{100}	String	*Not registered*	YES	
profile picture path	{100}	String	*Not found*		

5.5 Composition Viewpoint

The composition viewpoint describes how the framework is hierarchically structured into its constituent components. The following diagram address the design concern and describe the design elements involved and any implementation notes.

Our system works on Ubuntu or Linux Centos. For now, our database system uses MySQL. Since our server-side works with Node.js, Node.js must be installed on our system. Main program and database are going to work on Node.js. The data that comes from admin and interface will be checked with JavaScript. JQuery and html will be used for animations and for showing images, videos and texts. In

the far future after the project is evaluated we may have add-ons like news trends for filtering popular news freely roaming around our system.

After this section the components are mentioned more in detail.

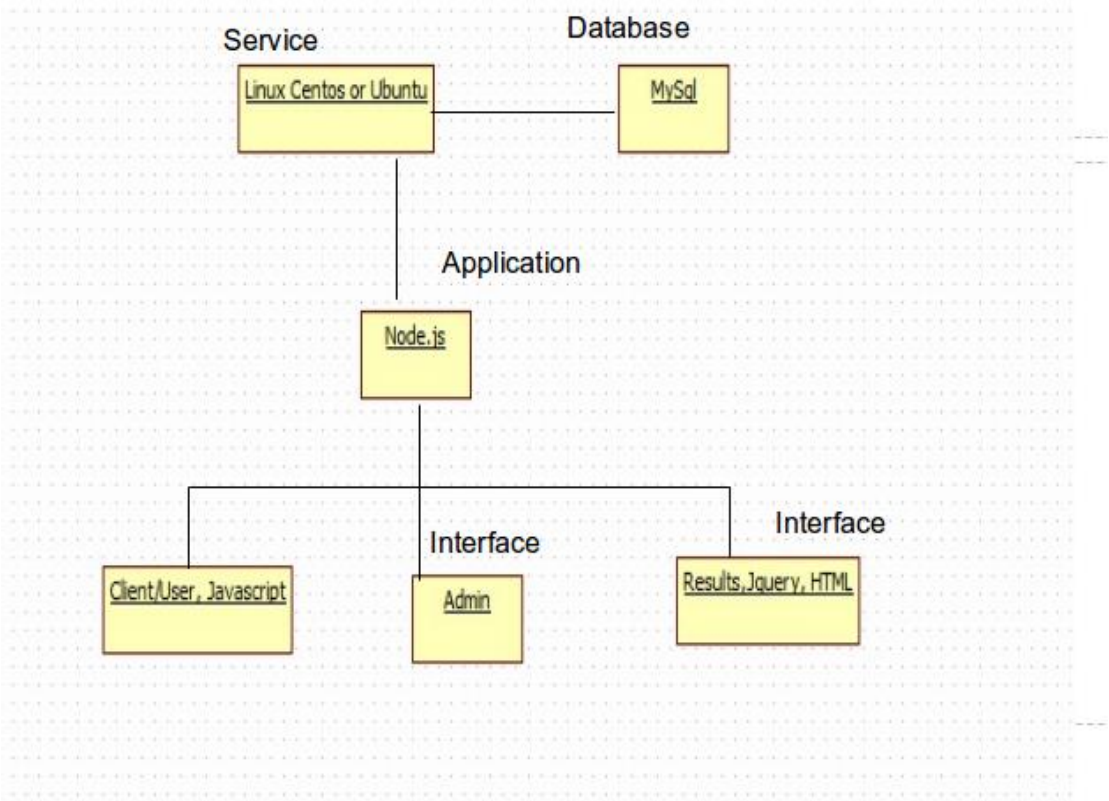


Figure 19 Software Components

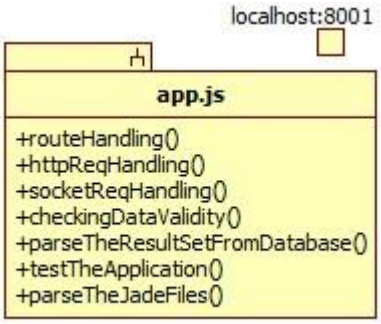


Figure 20 Main Program

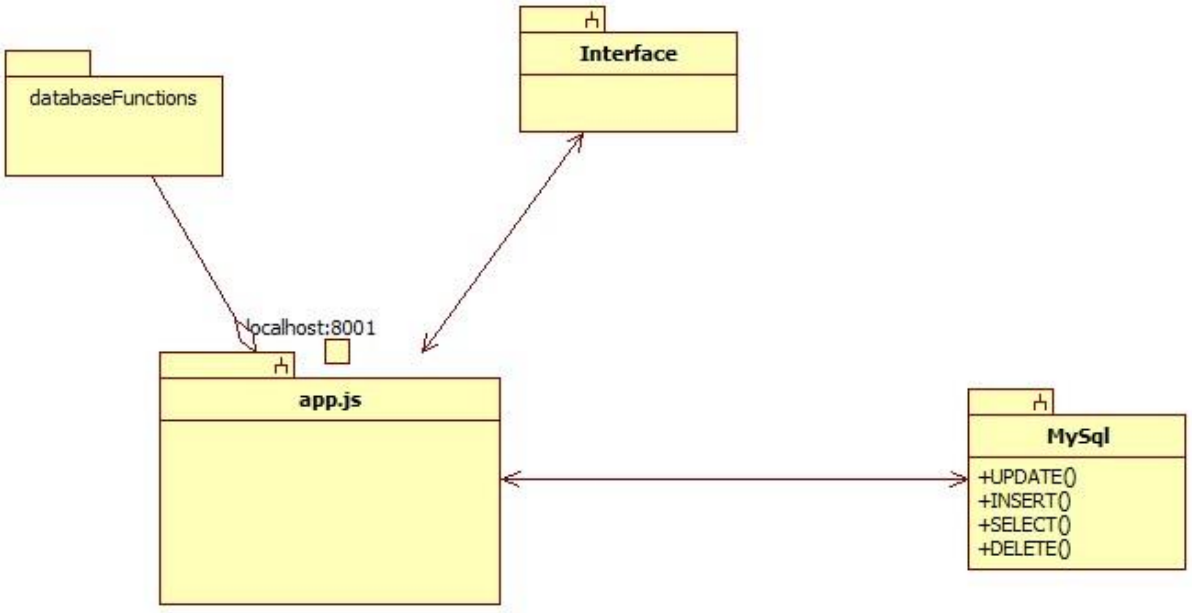


Figure 21 Database Interaction

5.5.1 Design elements

5.5.1.1 Design Entities

The node.js makes its developer to code everything from scratch excluding 3rd party packages and also from bright side a JSON files should be contained at root directory of an application to make node.js decide on which packages and modules must be imported after installing.

As framework we are using Express for handling http requests and route handling. It is wrapped in such a nice way to compensate the cons of JavaScript to satisfy coders' necessities on time constraints. As said before package.json handles all of the package and repository chores at once and for ever. Since at template and repository side there isn't much to record here but if we go deep into the Express side which is essential for every developer we come across some essential information of its structure.

From the site http://expressjs.com/ it describes itself as:

"Express is a minimal and flexible node.js web application framework, providing a robust set of features for building single and multi-page, and hybrid web applications."

"With a myriad of HTTP utility methods and Connect middleware at your disposal, creating a robust user-friendly API is quick and easy."

Say it has no dependency since NPM (node.js registry) handles all required chores like installing dependencies after reading the package.json. NPM is so-called registry where people can commit their Node.js wrap-ups for world-wide usage and getting bigger. In conclusion we can say so far everything is cost free.

5.5.1.2. Design Relationships

The entities (here mostly are packages) is displayed with a relationship below. As you can see app contains the NPM installed packages. Namely Node Package Manager (http://blog.nodejs.org/2013/11/26/npm-post-mortem/) NPM is a package manager which bundles all packages in a registry freely uploaded by all Node.JS developers globally. Its structure makes developers easily maintain their application's dependencies. The depicted figure below is somehow installed packages from NPM.

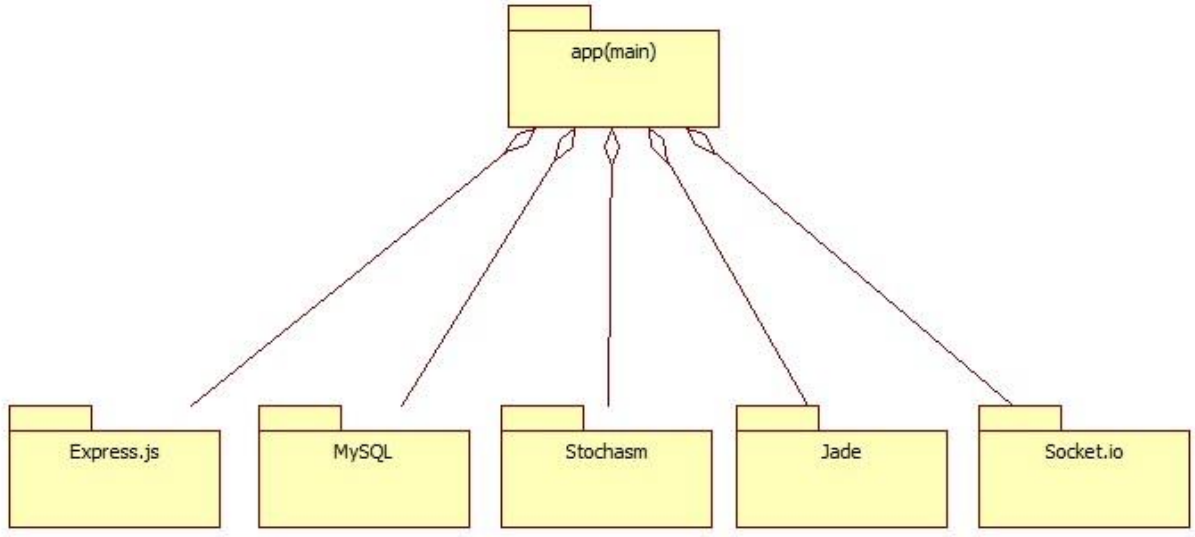


Figure 22 Design Relations

5.5.1.3. Functions as Entities

Since JavaScript doesn't owe any class structures besides the modules counted in above there are functions which has a valid input and output and so forth its own characteristic to be put as an entity. At least in our project there isn't much to count as entity other than because of the structure of heavily used JavaScript at both client and server side. These contained data is also informer of the attributes of the future data.

Name	**Brief Purpose**	**Location**

antiFloodAttack	Checks for the size of the input field object	app.js
listen	Listens to server	app.js
sockets.on	Listens for an emitted socket	app.js
loginpageFunction	Renders the loginpage.jade	/routes/loginpage.js
connect	Connects to DB with enlisted credentials.	/routes/databaseFuncs.js
insertFunc	Inserts a user	/routes/databaseFuncs.js
delete_userFunc	Deletes a user	/routes/databaseFuncs.js
update_passwordFunc	Updates the password	/routes/databaseFuncs.js
update_usernameFunc	Updates the username	/routes/databaseFuncs.js
add_follower	Inserts to “follower” table	/routes/databaseFuncs.js
add_friend	Inserts to “friend” table	/routes/databaseFuncs.js
select_n_users	Gets n (via parameters) users	/routes/databaseFuncs.js
update_emailFunc	Updates email	/routes/databaseFuncs.js
passWithEmail	Checks if the email is registered	/routes/databaseFuncs.js
passWithNick	Checks if the nick is registered	/routes/databaseFuncs.js
selectWithEmail	Selects a user with an unique email	/routes/databaseFuncs.js
selectWithNick	Selects a user with an unique nick	/routes/databaseFuncs.js
email_ExistsFunc	Checks whether user with the specified email is at the database	/routes/databaseFuncs.js
nickName_ExistsFunc	Check whether user with the specified nick exists	/routes/databaseFuncs.js

5.5.1.4. Database Tables Design

We followed a key-value pair approach where each pair contains actually like a subset contains elements from its parent. It increases the speed very much in favor. The essential detail here actually paths of posts are not the path of it but path of the metadata. This metadata includes some details like how many and which pictures does this post include or how the margin and posture over an html element should be in pixels.

Note that key value pairs are both primary key to prevent from addition of data in reverse order. For instance adding 44, 45 to a table which contains 45, 44.

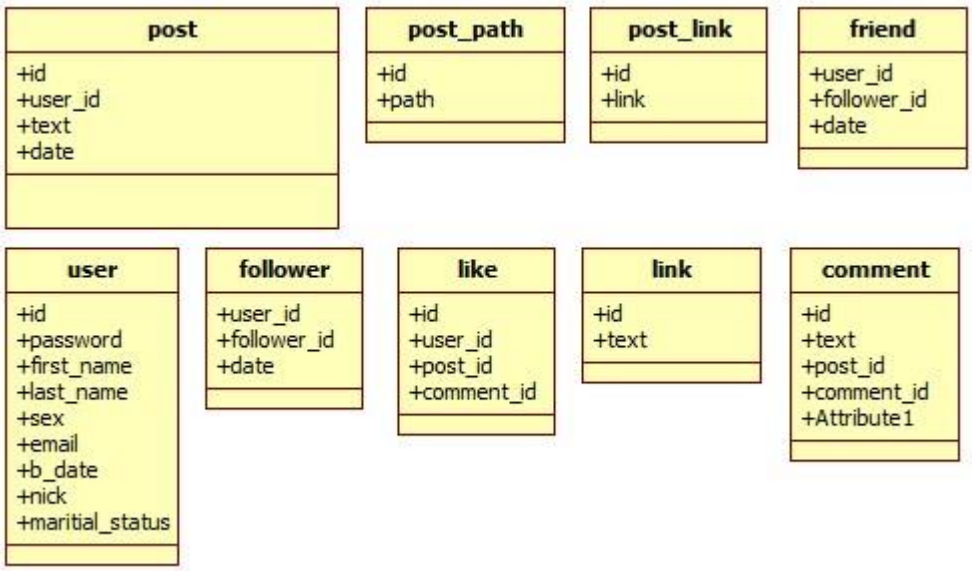


Figure 23 Database Table in Social Trends DB

5.6 Resource Viewpoint

The purpose of the resource viewpoint is to model the characteristics and utilization of resources in a design subject. Also what drains resources of our system and possible threads are included in this branch. Resource availability and utilization are also major points.

5.6.1. Possible Resource Drainers

First of all we have created our black hole strangely that is much capable of sucking much resources by enabling trial usage. At this scenario there has been people who wants to try a product but doesn't want to login or share his or her confidential data. May be s/he doesn't want to memorize millions of passwords so a person who is willing to experience our site just clicks on a trial account button and directly accepted by the system. When clicks log out or closes down the browser the session isn't stored any more.

The possible abuse are scripts using socket layers, just a tiny one, which also replicates itself and clone into system growing exponentially would cause servers stuck at big scale. Besides without any abuse it is still disadvantage to system with its extra burden.

The possible solutions are IP check and banning our freezing the IP for a state amount of time. Other is restricting the actions of the trial accounts.

On the second hand there isn't much kind of wrapped big data is stored in our server except images. Videos are still link and text are strings itself. There might be image compression tools in use. Also improving the connections' compression rates is a solution.

Other resource drainers aren't including that much abuse since they are ordinary tasks that every site must handle, clicking buttons, sending messages, writing to file.

5.6.2. Utilizing Static Data

At the back end we keep images in our file system rather than database since it would be a huge burden for a database and still not practical. When we get upload image through our system we are contemplating to compress it one way or another. This will prevent overhead causing RAM to work more and more.

The major point here picture upload is all about writing and reading files but the package we are using has very much drawbacks since it keeps the pictures in a directory called "tmp". After it sends to temp a new process starts this time to copy the content to server side. Currently we are using this module for easiness of use but in the future (Ref: https://github.com/felixge/node-formidable) most probably we will immigrate to a different high speed module.

5.6.3. Efficient Usage of the System

We have many options here. After observed some time of the system usage and ram drainage we are contemplating to enable or disable some plugins on both server and browser side. Memory leak should be covered after tedious work and coding must prevent unnecessary loops, setting a variables value more than one time in a loop etc.

The web developer doesn't approach to HTML and JavaScript like they are very efficient, contrarily especially HTML is recognized as having major performance drawbacks so all duty is up to coders.

Below system load is depicted in order after JQuery loaded to DOM user interactions start and it also causes some overhead between sockets on client and server side. We are completely dependent to enhancements on very low level packages namely NPM registry where we pull packages from. The graph below semantically depicts the decrease of speed the more components are loaded.

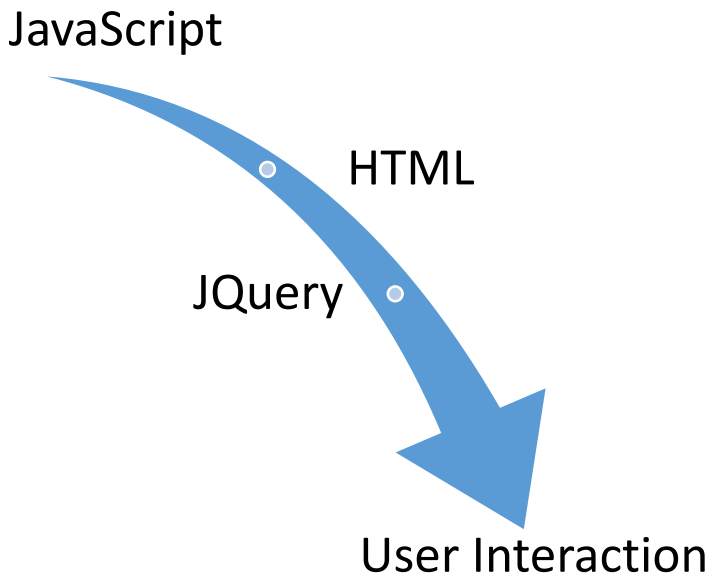


Figure 24 What DOM Parser Loads

5.6 Requirement Viewpoint

The requirement viewpoint describes (here) the key set of requirements derived from customer needs which forms the basis for the Social Trends. This part isn't created for repeating SRS all over but at more specific level what made us to think of this system and the requirements of other people who aren't yet satisfied with other social networks and somehow able to point their deficiencies.

Since Twitter and Facebook has improved gradually to their latest form today they have patched much of their major drawbacks both graphical and backend side but except one essential point: Filtering Data. It is such a major concern at last years because most of the data is not filtered globally backed up in mostly embedded devices and sensor and secondly at social networks. We are performing this tasks and filtering out three major trends which are text, video and image. The text trends are based on hashtags. Other strings not containing any hashtag are not filtered. (Extra information for this part)

The other dissatisfactions of users from Facebook for instance there is no dislike button to point out your irritation or reaction somehow. Think that you have this right, this would be huge relief! The interaction between dislike and the receiver is still off-topic but clearly the receiver won't share a non-appealing content after a while especially discovering what people dislike. If insists we have ready to denote most disliked trends.

The other drawbacks are the panels are not customizable both in Twitter and Facebook and still they are lacking giving the perception to user that his or her page belongs to her and makes custom changes. It is not a big surprise that Android gets the biggest share at market since its mantra is "Free

Software" [7] We will be allowing many customizations yet only one is drag and drop panels but in the future with developing on client side there may be resizable panels and for now there will be also removing panel and adding new panels. Since there are many magazine programs let you choose to filter the data according to your hobbies our project is unlikely this way because there will be for now limited amount of categories. Other difference is also unlike the other we don't filter the data from WWW but only at our servers. For instance you will be displayed from your friends which friend's post got the most dislike and focus on this point. It makes you completely stay connected to attract posts so to your friends and acquaintances and so on to the world.

Moreover we are offering in-post search what Twitter offers but Facebook does not. This allows users to look into posts more in depth or finding out what they are looking the time being.

6. One Year Plan

Until now we have covered SDD, SRS reports and some major components like business logic, very first database services and initial designs. Though they are less in words there hundreds lines of codes and very much time labor spent on the task. It is wise to share our future plans here in a well-ordered structure to ease the task of the audience of this document.

Frankly, it will take time to get over some user limit but in order to make plans and establish roads we need to approximately calculate predicted future work load, database tables' row counts in other word "size" so we can make some moves on basis of these statistics. The following content will assume our application is running on Heroku a popular Node.JS application server provider and using PostGRE SQL. Since the plans are estimated so far (yet the only solution since still barebones are in construction step) it is wise to use Heroku for now for statistical purposes only which has a lower level documentation than other service providers. Moreover knowing usage counts allows us to estimate the cost. Therefore we will be able to make more realistic, solid plans. This data which is depicted here nor binding neither restrictive.

On the other hand, though this branch is not completely leaved to some calculations also we should mention about our plans at architectural and component-wise level to "crystal clear" the path we are walking in, say roughly. The interpretation here are subject to change minimally yet still allowing the audience we have solid aim: Make people connect and socialize easier and unlikely from our future rivals like Facebook.

6.1 Database Workload

Heroku offer many options to its users. The one and the most important approach is "pay you grow." Besides that it has large amounts of software bundles that save a lot time. [3] With along the information at this site: https://devcenter.heroku.com/articles/heroku-postgres-plans as our basis point for now, say we are using standard tier with up to 1 hour down time per month it seems an appealing candidate. Within Standard Tier there are variety of options for memory and storage. Say we are using IKA having 512 GB database size at total, priced monthly $750. Since there is no row limitations we are free to play freely in this space provided.

The other point at this point calculating the user count. This is the reason why database workload ordered above other sub topics. Say a user has:

Name	Profile Pic Size	Post #1 Size	Post #2 Size	Post #3 Size
Anil Bektas	250 KB	200 KB	200 KB	200 KB

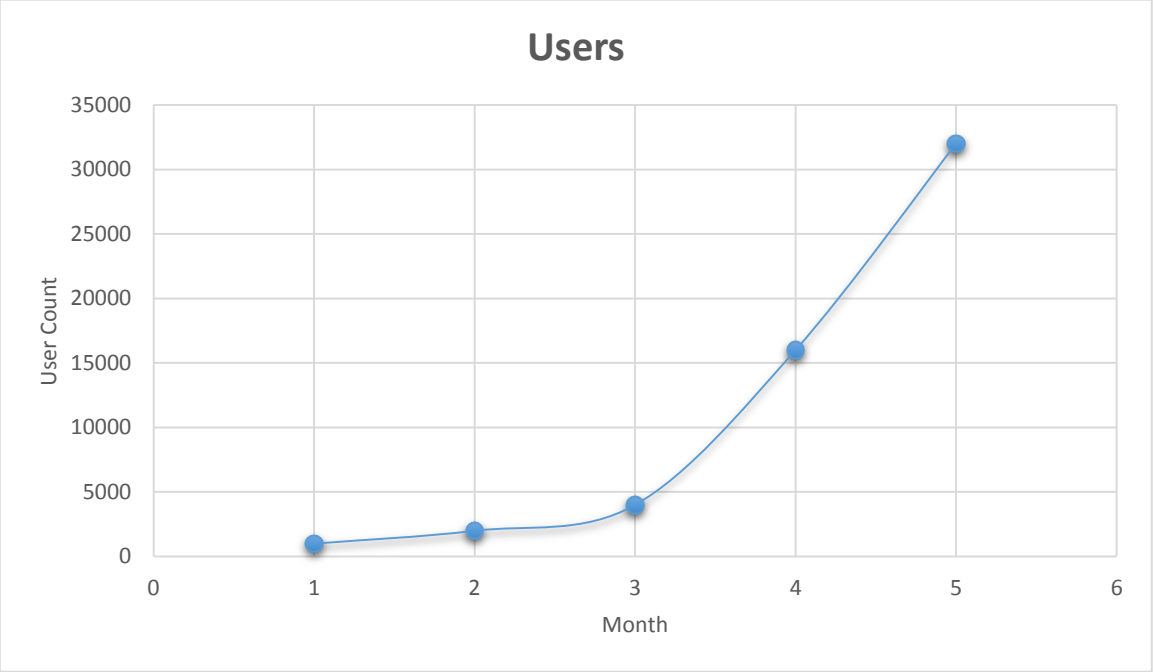
We should comment in that this table above depicting posts on daily frequency. At total user Anil Bektas has 250 KB plus 600 KB data burden daily. Assume all users are sharing this amount of data daily for 1 month and server side hasn't any burden.

512 gigabytes = 536 870 912 kilobytes

(536870912 – 250) / 600 = 894 784 users if the servers up time is just only 1 day.

894 784 / 30 = 29 826 users

The last operation is depicting the maximum amount of users after 1 month of 29 826 users are sharing regularly the database if fully out of capacity. Let's show the progress via a graph:



The graph shows that with IKA Tier if the scenario above is assumed to be true as use case after exponential growing of user during 5 months made our database system impossible to handle the more burden. Since the site is still in progress stage after first lunch at first ten months we are not contemplating 40 000 users signed up but about 30 000 users at maximum though it is still a nice number. Note that the exponential growth is chosen since we are thinking that our announcement campaign would lead to a viral growth. It seems to be the balanced case (close to worst case) since for instance at Facebook statistics look up [4] [5] 30 000 000 000 posts / (728 million/day * 30) concludes 1.37 posts per day created. Since we have 3 posts for each user per day if we scale our outputs to 3/1.37 times which is 2.1 the database is able to handle 58 000 users (2*29000) which seems to be stable at this first 10 month-phase since we are expecting about 30 000 users.

In conclusion the cost at this phase, besides the rests, is $750/month. The plan might be downgraded at first 10 month phase since it seems to be a bit luxury.

6.2 RAM and CPU

At first it should be stated that calculating CPU consumption rates is really hard, in other means it is impossible since we need to know how many active users interacting with the same server side components but since a dyno (namely working set in Heroku terminology) has 512 MB RAM we can conclude output CPU power.

Say, a user needs 10 KB of RAM through his or her active session since a dyno has 512 MB RAM 512 MB / 10 KB there can be 552 users active at the same time which means 552 dynos. Since the pricing web site is only calculating (Link: https://www.heroku.com/pricing#postgres) at most 150 dynos with a mind blowing cost ($5362) we have to interpolate other values to find out the output cost:

Dyno Count	**50**	**100**	**150**	**550**
Cost($)	1762	3562	5362	19762
Active Users At Same Time	50	100	150	552

In conclusion since the monetization is not still properly achieved the approximate dyno count of the system will be about 200 or 300. It will be conducting enough amount of operations for some time until we get an investment. We are concluding that with approximate cost like 7000 dollars plus 750 dollars business logic handling 200 sessions at the same time. Since the fiscal issues are inborn subject to change there might be downgrade for some months, like rather than using 200~dynos there might be even 20 dynos at $750/month for the time being. With total $1500 cost, it is still a big burden for young start-up students like us.

The cost analysis besides pushing us making realistic plans also giving a tip how and when we should scale our system or more precisely the system might go through some changes that we should implement in code-wise level at 2nd term for an effective, compressed data transfer. Above there are critical information about this topic.

6.3 Plans in Gantt Chart

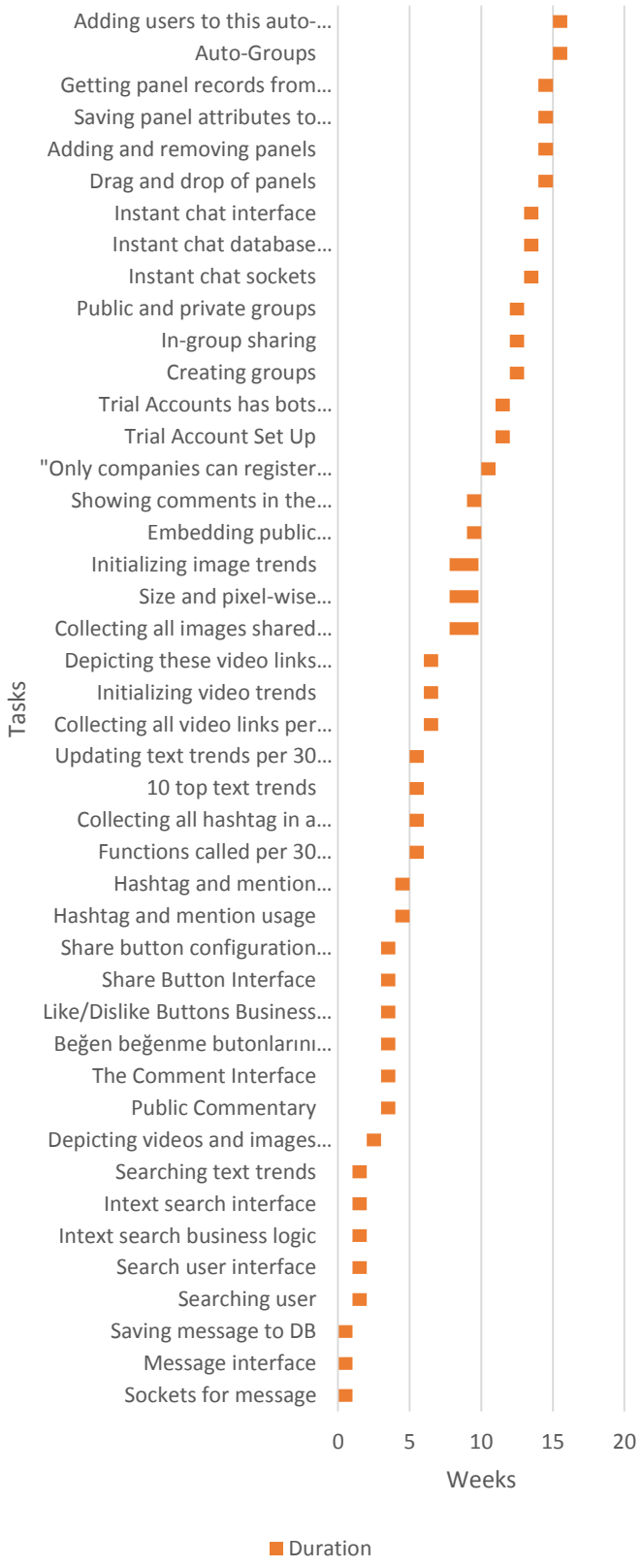
At this section there are future plans divided into 2 two subcategories: First term and the second term. There might be extra sub-tables for to be coherent with the margins and size of the document. The first table contains tasks mostly completed in previous weeks with their durations. Second type of table depicts our plans and drafts. These table nicely outline the gradual improvement of tasks and ease process of sketching what should we do at what day or month.

The first chart know as depicting the whole first term task the last 3-4 tasks are still in progress.


Term 1 Grantt Chart
New Viewpoints
Integrating the Picture Uploader
Presentation and Rehersal
Iteration of the Documents
Displaying Posts
User Picture Upload
Sending Posts
Interface of "Post" div
Interface of "User" div
Sockets on Homepage
Bootstrapped scrollable div
Suggesting random users
Python Script to Create Tables
LocalStorage(Cookie)
Random Background
First Bootstrap
Login/Home Page Design
Active Jquery Usage
Adding Friends
More Database Integration
Viewpoints
Enhancing SDD and Server
Data Transfer Controls
Interactive boxes
QT Search
Combining Database To Server
Sockets.io
SDD (1,2,3,4)
Database Calls
Login Page
Picture Uploader
Application Server
Literature Research
Market Research
Tasks
0
2
4
6
8
10
12
14
16
18
Weeks
Duration

From the first chart you can observe that our work output has grown exponentially while the durations are strictly obeyed in other terms stayed the same. At the end of the first term, before the demo we are predicting to nicely craft the login and home pages completely, showing up Facebook-style window area when clicked on post, suggesting friends and sending posts. Besides fully completed log in and sign up forms are ready to be enhance to their last stage at server side: Confirmation e-mail. Bear in mind that some tasks more or less more smaller but critical are not be placed to the graph since they were some kind of must-have behind the tasks stated, for instance when adding friends also at business logic database transactions must be made. These small details are left to audience's own recognition but here we are reminding nevertheless.

The second term plans are depicted below. In summary about June we will have created the systems' major components and what we outlined before. From bottom to up graph is getting more and more close to the last deadline. Unfortunately this page's below content is unintentionally left blank because the diagram isn't split.


2nd Term Tasks
Adding users to this auto-...
Auto-Groups
Getting panel records from...
Saving panel attributes to...
Adding and removing panels
Drag and drop of panels
Instant chat interface
Instant chat database...
Instant chat sockets
Public and private groups
In-group sharing
Creating groups
Trial Accounts has bots...
Trial Account Set Up
"Only companies can register...
Showing comments in the...
Embedding public...
Initializing image trends
Size and pixel-wise...
Collecting all images shared...
Depicting these video links...
Initializing video trends
Collecting all video links per...
Updating text trends per 30...
10 top text trends
Collecting all hashtag in a...
Functions called per 30...
Hashtag and mention...
Hashtag and mention usage
Share button configuration...
Share Button Interface
Like/Dislike Buttons Business...
Beğen beğenme butonlarını...
The Comment Interface
Public Commentary
Depicting videos and images...
Searching text trends
Intext search interface
Intext search business logic
Search user interface
Searching user
Saving message to DB
Message interface
Sockets for message
Tasks
0
5
10
15
20
Weeks
Duration

7. The Future

Since the social network sites generally attain too much data flow so does our one. In the future where our databases are not going to handle the terabytes of data any more. In order to keep system alive and fast enough we may need HADOOP systems. From where first Google was found with an article which allows the data to break apart and processed on different machines until Apache adapt into their trademark there is a bias of big companies to HADOOP systems to increase their efficiency, profit debt and speed over "big data". [6]

There has been a seminar at this topic in METU Computer Engineering in December 2013 by a company called OBSS. For instance Walmart a multinational retail corporation that runs of chains of market implemented a software solution. This program searches and interacts with the company's very large data sets to check whether it is rainy at certain location and send customers at this location a message like "Hey, we have raincoat, would you like to buy?" and achieved to increase their profits. Other ways of likely implementations may be started to increase efficiency of the web site.

8. Conclusion

In conclusion we have provided viewpoints to revive the system in the eyes of the audience with along using many graphical tools, diagrams and shapes. This document is eligible to give guidance to testers, designer and coders themselves enabling their coordination under single vision. The document also includes interface design besides the low level detail of the Social Trends. Hopefully with the help of these documents we will be walking on a path which has its borders clearly aligned and stroke to move the social networking one step up.

9. References

[1] IEEE. IEEE Std 1016-2009 IEEE Recommended Practice for Software Design Description. IEEE Computer Society, 2009.

[2] "Social Trends" Software Specification Requirements (SRS) Document, 2013. [3] For UML sources: http://www.tutorialspoint.com/uml/index.htm

[3]

(n.d.). Retrieved from https://addons.heroku.com/

Add powerful functionality to your apps with ease. (2013). Retrieved from Heroku: https://addons.heroku.com/

[4]

(n.d.). Retrieved from http://expandedramblings.com/index.php/by-the-numbers-17-amazing-facebook-stats/#.UrMeePRdV_M

By the Numbers: 104 Amazing Facebook Statistics. (2013). Retrieved from Expanded Ramblings: http://expandedramblings.com/index.php/by-the-numbers-17-amazing-facebook-stats/#.UrMeePRdV_M

[5] (n.d.). Retrieved from http://blog.kissmetrics.com/facebook-statistics/

Facebook Statistics. (2013). Retrieved from Kiss Metrics: http://blog.kissmetrics.com/facebook-statistics/

[6] (n.d.). Retrieved from http://en.wikipedia.org/wiki/Apache_Hadoop

Apache Hadoop. (2013). Retrieved from Wikipedia: http://en.wikipedia.org/wiki/Apache_Hadoop

[7] (n.d.). Retrieved from http://www.gsmarena.com/android_worldwide_marketshare_crosses_80_for_the_first_time-news-7171.php

Android Global Market Shares Now Exceeds 80%. (2013) Retrieved from GSM Arena: http://www.gsmarena.com/android_worldwide_marketshare_crosses_80_for_the_first_time-news-7171.php

10. Change Log

Changes to SDD after revision are listed as follows:

- ✓ Cover page got renewed.
- ✓ Figures were named and numbered.
- ✓ Preface section was removed.
- ✓ In section 2, the definitions of the terms were written in a table to be clearer.
- ✓ Section 3. Conceptual Model was elaborated more upon in order for it to be more clear and explanatory.
- ✓ In section 3.1.2 one table was added to show specific technology and packages we used in our system.
- ✓ Section 5 named Design Description in the previous SDD document was changed to section 5. Viewpoints in the new SDD document.
- ✓ The text in section 5.1 Context Viewpoint was elaborated more on to be more specific and also the figures for personal actions, interactions with others, miscellaneous actions and also Admin use cases were added for more specifications.
- ✓ Section 5.2 Interaction Viewpoint was elaborated more on, and Interaction flowchart and canonical cycles were added for signing up to logging out stages.

- ✓ A new sub-section; 5.2.1 was added to describe Design Elements and Constraints and figures were added to explain features better.
- ✓ Unnecessary figures from section 5.4 was removed and the remaining figures were elaborated on.
- ✓ Sub-section 5.4.1 was added to explain User Interfaces with some explanations and also more and new illustrations.
- ✓ Sub-section 5.4.2 was added to describe valid inputs and outputs for the log-in, sign-up and homepage stages. Each were described clearly in tables.
- ✓ Sub-section 5.4.3 was added to describe Data Attributes and a table was added to it for more clarification.
- ✓ Section 5.5 was elaborated more on, and also new illustrations were added for more clarification.
- ✓ Sub-section 5.5.1 was added to explain the design elements. This section has its own sub-sections with explanations and illustrations for each of them.
- ✓ New section 5.6 was added to describe Resource Viewpoint. This section also has its own sub-sections with explanations and illustrations for more clarification.
- ✓ New section 5.7 Requirement Viewpoint was added and explained.
- ✓ New Section 6 was added to give the One Year Plan for the project. It has its own sub-sections and each are explained and illustrated for better understanding.
- ✓ New section 7 was added to explain our future plans for the system and its features.
- ✓ New section 8 was added to give a Conclusion about the document and the project.
- ✓ New section 9 was added to give the list of references we have used, and the References section from section 1 of the old SDD document was removed.