**T.Y. B. Sc Semester VI (Computer Science)**

**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| **SR.NO** | **CONTENTS** | | **Page. No** |
| **1.** | **Introduction** | | 2 |
|  |  | Project Profile | 3 |
| **2.** | **System Introduction** | | 6 |
|  |  | System Definition, Objective and scope | 7 |
|  |  | Hardware & Software Requirements | 8 |
| **3.** | **Requirement Analysis & Modeling** | | 9 |
|  |  | Expected working of system | 10 |
|  | Data Flow Diagrams (DFD) | 11 |
|  | Process specification | 14 |
|  | Data Dictionary | 18 |
| **4.** | **Design** | | 22 |
|  |  | ER Diagram & Database Design | 23 |
|  | System Architecture | 34 |
|  | System Flowchart | 35 |
|  | Form & Report Designs (Screen Layouts) | 36 |
| **5.** | **Coding** | | 45 |
|  |  | Coding Approach/style | 46 |
|  | Code Snippets | 48 |
| **6.** | **Testing (**Test Cases, Test Data & Results ) | | 61 |
|  |  | Testing Data Inputs ( Data Validations) | 62 |
|  | Testing Operations (Actions on each form) | 63 |
|  | Testing Correctness and Completeness | 64 |
| **7.** | **New Tools/ Technologies learned/used** | | 65 |
| **8.** | **System Limitations/Restrictions and Dependencies/Constraints** | | 68 |
| **9.** | **Future Enhancement & Opportunities** | | 70 |
| **10.** | **Bibliography & References** | | 72 |

**Introduction**

**Project Profile**

What TimeGod is:

TimeGod is a new-era time-table creation website, an improvement over the already exisiting time-table creating websites.  
Most of the services out there are, in their own terms specifically targeted towards their own customers,   
TimeGod, is targeted towards faculties who have a hard time creating a timetable that has multiple constraints involved- because face it- it’s nerve-wrecking!

What to expect of TimeGod:

While other web-services provide you the freedom to use a drag & drop GUI to decorate your time-table with pretty colors, TimeGod not only does that but also makes your time-table for you.

One can expect:

* Multiple division-processing within multiple-batch processing.
* The freedom of assigning number of hours per required subjects.
* The ability to assign respective faculty to their subjects and take care that their schedule don’t clash at two places.
* The ability to give any subjects a preference over another.
* A natural-formation of time-table that follows the format of heavy mid week-days (Wed, Thu), gradually thinning out over the weekends and the black days (aka Mon, Tue).
* A simple one-page website holding a 7-step wizard to artfully indite your perfect time-table.
* The pleasure of coming back whenever you want and editing the time-table as you wish.
* Oh and a direct, ready-to-print .docx file on your final step.

**System Introduction**

**System Definition**

TimeGod is a time-table creation website that heavily relies on javascript and servlets to tailor you the perfect ergonomic time-table for your instituition. It aims at delivering ease of use, swiftness and high accessibility to make the experience as smooth as possible. It also handles various parameters that take into account that faculties don’t have clashing lectures or a certain faculty doesn’t have to take two lectures at the same time; ergo, diminishing the nerve-wrecking headache it is to manage all that information. Let TimeGod worry about it.

**System Scope**

* Currently, none of the time-table creation sites provide an intelligent backend that could guide you with your assessments of the required time-table.
* Even more so, none of them have a dedicated backend that works completely upon delivering a mid-out thinning week-plan that is not only student-friendly but also faculty-friendly and hence reduces human-effort friction.
* This is a major missing dent since technology now than ever is highly capable of eliminating tedious routines which could swiftly be handled with assigning tasks to the computer itself.

**System Objectives**

* What TimeGod intends to do is eliminate the entire decision making process that people have to go through so that faculties and students are not put into considerable load whence the time-table isnt created.
* Not only that but, creating a time-table in a wizard format proves to be more useful as it is not only a mere 7-step wizard but also the simplest one could think of.
* TimeGod takes into account your entire data- ranging from number of courses you run, to number of faculties that will be taking a subject in that division in that batch in that course and make sure that no faculties ever clash their timings or have to be at the same time at two different places.
* It also supplies to you a printable .docx file that is sent to your desired e-mail address so that you may print it express, pin it and so that you can easily find it when you lose it.

**System Environment Description**

The efficient hardware and software requirements to run this website smoothly,  
would be as follows:

Hardware Requirements:

* Processor: Intel i3 6th Gen.
* Ram: At Least 4GB DDR3.
* Graphics Card: 512MB (Onboard or otherwise).
* HDD: Minimal. (SSD recommended.)

Software Specifications:

* Front End: JSP
* Back End: MySQL
* OS: Windows 10

**Requirement Analysis  
and  
Modelling**

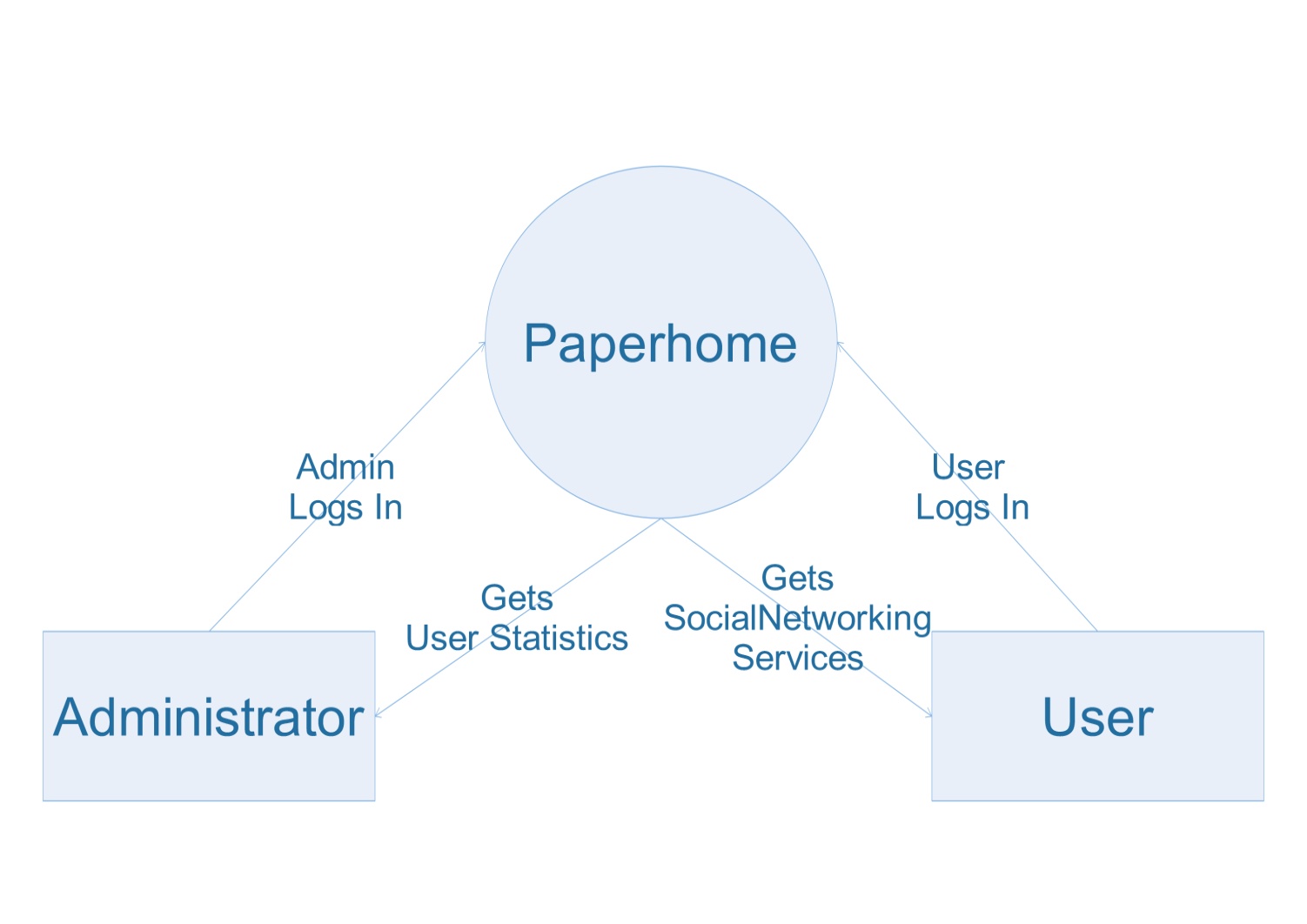
**Expected Working of System**

**Users could:**

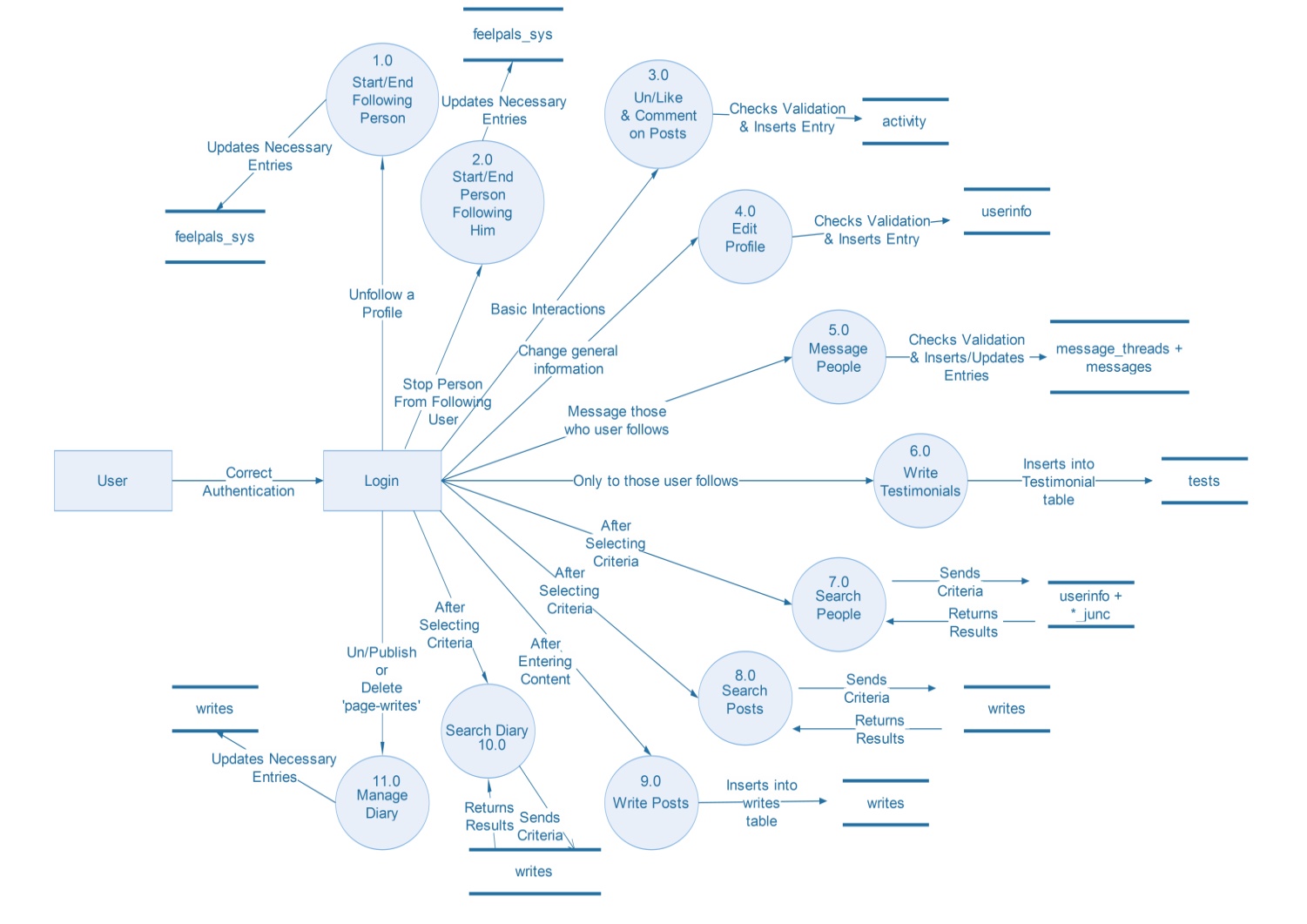
* Create time-tables through a 7-step wizard.
* Download a printable .docx of the time-table.

**Data Flow Diagrams**

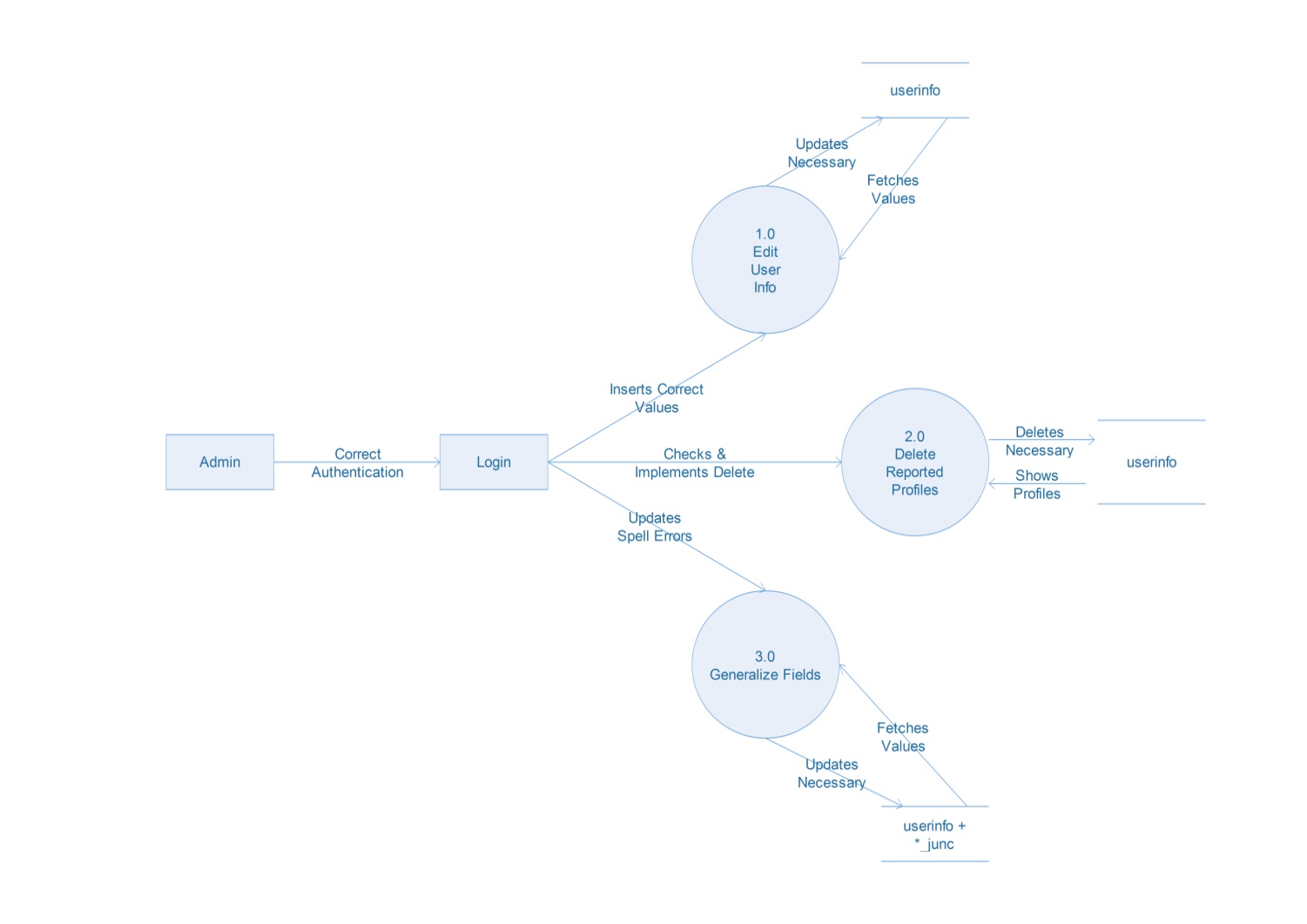
**0th Level (Context) DFD:**

****

**1st Level (User) DFD:**

****

**1st Level (Administrator) DFD:**

****

**Process Specification**

**User Side:**

**Process 1.0:**

|  |  |
| --- | --- |
| **Name** | Start/End Following Person |
| **Input** | User will click on the un/follow button on the person’s profile. |
| **Output** | Corresponding entries in the ‘feelpals\_sys’ table and/or the ‘req’ as well as ‘activity’ table will be made. |
| **Description** | This process aids in helping people to un/follow a certain person. |

**Process 2.0:**

|  |  |
| --- | --- |
| **Name** | Start/End Person Following Him |
| **Input** | User will click on the remove follower button or the accept request button displayed on various pages. |
| **Output** | Corresponding entries will be made in the ‘req’ table or the ‘feelpals\_sys’ & ‘activity’ tables. |
| **Description** | This process aids in helping people get rid of a certain follower or accept someon’es follow request. |

**Process 3.0:**

|  |  |
| --- | --- |
| **Name** | Un/Like & Comment on Posts |
| **Input** | Users will click on the ‘like’ button or the ‘comment’ button after entering a comment. |
| **Output** | Corresponding entries will be made in the ‘activity’ & ‘comments’ table. |
| **Description** | This aids in helping users like a ‘page-write’ or comment on one. |

**Process 4.0:**

|  |  |
| --- | --- |
| **Name** | Edit Settings |
| **Input** | Users can input values for various fields related to their profile information. |
| **Output** | This in turn, will update their information in the ‘userinfo’ table and other related tables as necessary. |
| **Description** | This processs aids user to update their profile in an easy format. |

**Process 5.0:**

|  |  |
| --- | --- |
| **Name** | Message People |
| **Input** | Users can enter message content on selected pages and send those messages to people. |
| **Output** | A new thread will be created if the users haven’t interacted previously or the thread had been deleted by one of the users. The messages will be added related to their unique thread IDs in the ‘message\_threads’ table. |
| **Description** | This aids user in sending and recieiving messages from other users. |

**Process 6.0:**

|  |  |
| --- | --- |
| **Name** | Write Testimonials |
| **Input** | Users can input content into their testimonials for other users and click the send button. |
| **Output** | This will tag the user who sent it and to whom and the testimonial will be then visible on the reciever’s profile. |
| **Description** | This aids users in sending and receiving a testimonial. |

**Process 7.0:**

|  |  |
| --- | --- |
| **Name** | Search People |
| **Input** | The user can enter criterias and/or select predefined filters for searching. |
| **Output** | The module will return valid results that match the user’s filters and criterias. |
| **Description** | Helps find the perfect person user wants to find by taking into consideration the person’s mode feelings, likes and dislikes, interests and hobbies. |

**Process 8.0:**

|  |  |
| --- | --- |
| **Name** | Search Posts |
| **Input** | The user can enter criterias and/or select predefined filters for searching. |
| **Output** | The module will return valid results that match the user’s filters and criterias. |
| **Description** | Helps find the exact kind of ‘page-writes’ the user is looking for by taking into consideration the user’s criterias and filters applied onto the search engine. |

**Process 9.0:**

|  |  |
| --- | --- |
| **Name** | Write Posts |
| **Input** | The user can write content of his/her day, add a date, assigned feeling, assigned hashtags and a couple of extra tags. |
| **Output** | This entry will be saved into the ‘writes’ table and will be visible in the user’s MyDiary whenever the user wishes. |
| **Description** | The process helps in adding ‘page-writes’ to the user diary. |

**Process 10.0:**

|  |  |
| --- | --- |
| **Name** | Search Diary |
| **Input** | The user can apply various filters and enter his/her required criterias in order to find the kind of posts he wishes to see/review or edit. |
| **Output** | The database will return results based on the user’s criterias from the ‘writes’ table if available. |
| **Description** | Aids in the process of re-living diary moments or simply find a lost note. |

**Process 11.0:**

|  |  |
| --- | --- |
| **Name** | Manage Diary |
| **Input** | The user can manage the ‘page-writes’ and toggle their visibility on the profile. And/or delete- ‘burn’ the ‘page-write’ if he/she wishes. |
| **Output** | Necessary changes are made in the ‘writes’ table. |
| **Description** | Helps managing user diary. |

**Administrator Side:**

**Process 1.0:**

|  |  |
| --- | --- |
| **Name** | Edit User Info |
| **Input** | The administrator checks the fields per user displayed on his side and edits those which seems necessary implemented through an ‘edit’ button click. |
| **Output** | The entries in associated tables are changed and the user is notified of it. |
| **Description** | Helps removing duplicate data and spell errors. |

**Process 2.0:**

|  |  |
| --- | --- |
| **Name** | Delete Reported Profiles |
| **Input** | The reported profiles (by user) are reviewed by the administrator and if the administrator seems fit he has the power to delete a user profile though a button click guided by a confirmation prompt. |
| **Output** | Various tables are affected and the user profile is deleted without leaving any traces anywhere. |
| **Description** | Aids in removing spam profiles or duplicate profiles. |

**Process 3.0:**

|  |  |
| --- | --- |
| **Name** | Generalize Fields |
| **Input** | The interests/hobbies & favourites table data is displayed and the user can modify them by either replacing duplicate entires or deleting non-eligible entries or by assigning the entries to different junctions. Each have their own button event. |
| **Output** | The data fields that are modified, replaced or removed affect their corresponding data tables. |
| **Description** | This process is very important as it regulates data flow of the search algorithm and provides a smooth-flowing search experience. |

**Data Dictionary**

**Table Name**: userinfo

|  |  |
| --- | --- |
| Data | Information of registered users. |
| Use | Used in generating profiles and associating user-data. |
| Description | Contains name, username, password, and other basic information that is taken during registration process. Also contains the privacy levels as values. Is also regularly updated as and when user logs in. |

**Table Name**: writes

|  |  |
| --- | --- |
| Data | Stores the ‘diary pages’ that a user writes. |
| Use | Used in generating the complex ‘writes’-template used throughout the website. |
| Description | Contains the content, date/time, associated feeling, hashtags and profile ID of the user to whom the write originally belongs.  Also contains the link to the comment-table and photo attachments. |

**Table Name**: tests

|  |  |
| --- | --- |
| Data | Stores the testimonials a user writes for another user. |
| Use | Used in profile-generation. |
| Description | Contains the content of the testimonial, rating and the date-time written. Is recognized with ‘from’ and ‘to’. |

**Table Name**: message\_threads

|  |  |
| --- | --- |
| Data | Maintains the thread IDs of the conversation between two users. |
| Use | Used in thread-generation on the ‘messages’ page. |
| Description | Contains the profile IDs of two users participating in a thread and assigns a thread ID (auto-number) which is then used to refer in the messages table. |

**Table Name**: feelpals\_sys

|  |  |
| --- | --- |
| Data | Maintains records of who follows who. |
| Use | Used to establish a system of ‘following’ and ‘followers’- this helps in monitoring privacy-related restrictions. |
| Description | Contains two fields- ‘group\_to\_following’ & ‘group\_to\_followers’- upon grouping any one of them, a follow-list or followers-list can be obtained. |

**Table Name**: comments

|  |  |
| --- | --- |
| Data | Stores all the comments made on the website. |
| Use | Used in comment generation over ‘page writes’ the user writes. |
| Description | Is reffered with writes\_id of ‘writes’ table-in order to maintain which comments belong to which write and so on. (Is obviously date-time stamped.) |

**Table Name**: activity

|  |  |
| --- | --- |
| Data | Stores all the activities a user does for all users. Activities include:   * Writing a new page. * Updating profile. * Liking a post. * Commenting on a post. * Start to follow another user. * Publish a post. |
| Use | Used in generation of ‘Recent Activity’ module found on each user’s home page. |
| Description | Contains date-time stamps, the doer and doee of the participating activity. |

**Table Name**: req

|  |  |
| --- | --- |
| Data | Stores all the ‘follow’ requests sent by an user to another. |
| Use | Used in generations of ‘follow requests’ on the ‘activity’ page. |
| Description | Contains date-time stamp along with a reference to profile IDs of the user who sent it and to whom. Also whether it is accepted or not. |

**Table Name**: messages

|  |  |
| --- | --- |
| Data | Stores all the messages sent all over the website between users. |
| Use | Used in conversation-generation on the ‘messages’ page. |
| Description | Is referred by a thread ID from the ‘message\_threads’ table which helps sort the messages to their belonging conversation. Date/time-stamped. |

**Table Name**: session\_logs

|  |  |
| --- | --- |
| Data | Stores login sessions along with IP address and browser name & version. |
| Use | Used in generating session logs in the ‘settings’ page. |
| Description | Contains IP addresses, date-time stamps, profile IDs and more regarding logins occurring. |

**Table Name**: music\_data, movies\_data, books\_data, food\_data, tv\_data, team\_data, places\_data.

|  |  |
| --- | --- |
| Data | Stores the data pertaining to various fields of favorites. |
| Use | Used in profile-generation and queried user search. |
| Description | Contains data-names for each table and relates to a specific ID which is then used in the junction tables (below). |

**Table Name**: music\_junc, movies\_junc, books\_junc, food\_junc, tv\_junc, team\_junc, places\_junc.

|  |  |
| --- | --- |
| Data | Stores the ID from reffered data-table (above) with associated profile IDs. |
| Use | Used in profile-generation and queried user search. |
| Description | Contains profile IDs associated with field IDs so as to relate whose likes are whose. |

**Table Name**: int

|  |  |
| --- | --- |
| Data | Stores the interests an individual has. |
| Use | Used in profile-generation and queried user search. |
| Description | Contains the fields of interests that every user has entered while registration. |

**Table Name**: int\_junc

|  |  |
| --- | --- |
| Data | Stores the ID from reffered data-table (above) with associated profile IDs. |
| Use | Used in profile-generation and queried user search. |
| Description | Contains profile IDs associated with field IDs so as to relate whose likes are whose. |

**Table Name**: inf\_fav

|  |  |
| --- | --- |
| Data | Stores the hobbies an individual has. |
| Use | Used in profile-generation and queried user search. |
| Description | Contains the fields of hobbies that every user has entered while registration. |

**Table Name**: int\_fav\_junc

|  |  |
| --- | --- |
| Data | Stores the ID from reffered data-table (above) with associated profile IDs. |
| Use | Used in profile-generation and queried user search. |
| Description | Contains profile IDs associated with field IDs so as to relate whose likes are whose. |

**Table Name**: picture\_library

|  |  |
| --- | --- |
| Data | Stores the picture information the user uploads. |
| Use | Helps fetching photos in the Photo Library. |
| Description | Contains photo IDs associated with profile IDs so as to retrieve whos photos are whose. |

**Table Name**: block\_list

|  |  |
| --- | --- |
| Data | Stores the profile IDs of the blockee and blocker. |
| Use | Used in managing profiles so that blocked/blocker people don’t engage as expected. |
| Description | Contains profile IDs associated with profile IDs so as to relate who blocked who. |

**Table Name**: livesessions

|  |  |
| --- | --- |
| Data | Stores the host ID, content and attachments that are required to hold a Live Session. |
| Use | Used in managing and hosting the ScreenBoard content in LiveJoin. |
| Description | Contains profile IDs of who hosts the sessions and manages related content. |

**Table Name**: bit\_data

|  |  |
| --- | --- |
| Data | Stores bit ID, bit content, sender ID, LiveJoinTM session ID & date written. |
| Use | Used in managing the LiveJoinTM discussion groups. |
| Description | Contains profile IDs of who sends bits for which session and what they contain. |

**Table Name**: library\_data

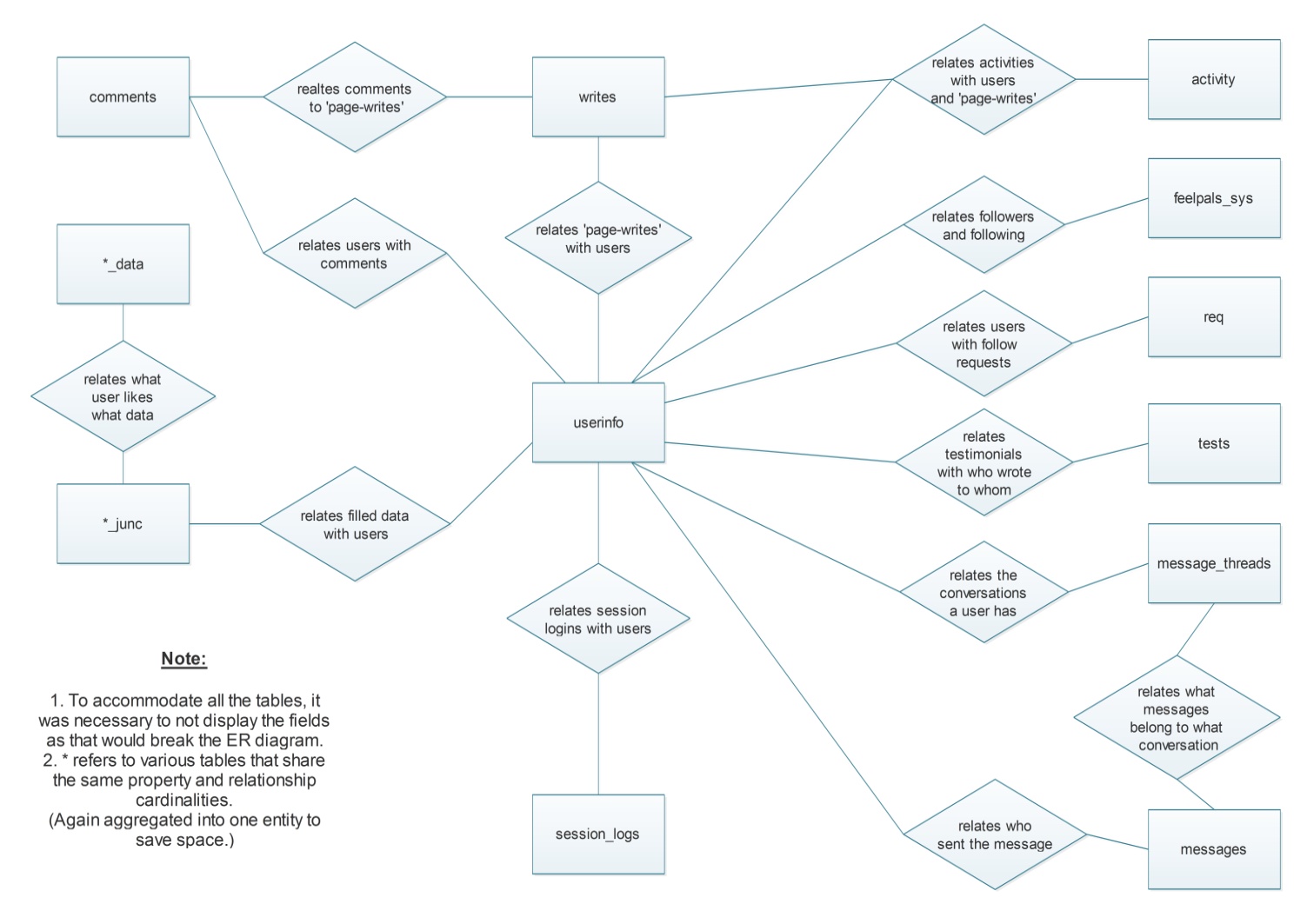
|  |  |
| --- | --- |
| Data | Stores the book ID, book name and path where the book is stored. |
| Use | Used in managing, searching and displaying book at Library.aspx. |
| Description | Contains book data that is vital to the library section. |

**Design**

**ER Diagram**

The entity relationship model is a high level data model. It is based on a perception of a real world that consists of a collection of basic objects, called entities, and of relationship among these objects. It was developed to facilitate database design by allowing specification of an enterprise schema, which represent the overall logical structure of a database.

**Entity:** An entity is an object that has its existence in the real world. It includes all those “things” about which data is collected. An entity may be a tangible object such as a student, a place or a part. It may also be non-tangible such as an event, a job title or a customer account.   
 **Attributes**: Attributes are units that describe the characteristics or properties of entities. In a database, entities are represented by tables and attributes by columns. They are drawn in elliptical shapes along with the entity rectangles.



**Database Design**

**Table Name**: userinfo

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | profile\_id | | |
| **Foreign Key** | <None> | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| profile\_id | AutoNumber | Not Null | Auto-numbering profile IDs which act as reference to a profile. |
| fname | Text | Not Null | First name of user. |
| lname | Text | Not Null | Last name of user. |
| dob | Date/Time | Not Null | Date of birth of user. |
| city | Text | Not Null | City of user. |
| country | Text | Not Null | Country of user. |
| sex | Text | Not Null | Gender of user. |
| email | Text | Not Null | E-mail address of user. |
| uname | Text | Not Null | Preferred & assigned username of user. |
| cell\_no | Text | Not Null | Phone number of user. |
| join\_date | Date/Time | Not Null | The date the user registers for the website. |
| about\_me | Memo | Not Null | Basic textual portfolio a user updates. |
| mode\_1 | Text | Not Null | User’s most-felt feeling. (Based on the feelings associated with his/her ‘page writes’.) |
| mode\_2 | Text | Not Null | User’s 2nd-most-felt feeling. (Based on the feelings associated with his/her ‘page writes’.) |
| is\_reported | Yes/No | Not Null | Tag if user’s profile is reported by another user. (Admin purposes.) |
| photo\_priv | Number | Default Value: 1 | Sets the photo-library visibility level for the user profile. |
| writes\_priv | Number | Default Value: 1 | Sets the writes-library visibility level for the user profile. |
| request\_priv | Number | Default Value: 1 | Sets the request/message option visibility for that user. |
| dp\_url | Text | Default Value:  img\avatar-profile.png | Sets the default display picture avatar and handles the paths to corresponding display pictures. |

**Table Name**: feelpals\_sys

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | <None> | | |
| **Foreign Key** | group\_to\_following, group\_to\_followers | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| group\_to\_following | Number | Not Null | Stores profile IDs. Upon grouping by a profile ID- returns the users that this user is following. |
| group\_to\_followers | Number | Not Null | Stores profile IDs. Upon grouping by a profile ID- returns the users that follow this user. |

**Table Name**: writes

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | writes\_id | | |
| **Foreign Key** | profile\_id | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| writes\_id | AutoNumber | Not Null | Refers to each ‘page-write’ every user writes. |
| content | Memo | Not Null | Contains the content a user writes into the ‘page-writes’. |
| hashes | Text | Not Null | Contains the hashtags a user associates with his written ‘page-write’. |
| published | Yes/No | Not Null | Determines whether the ‘page-write’ is up for public display or not. |
| date\_written | Date/Time | Not Null | Stores the date and time the user associates with his writes. NOTE: Does not store the actual date and time the write was written. That’s stored in ‘activity’ table. |
| feeling\_type | Text | Not Null | Stores the feeling the user associates with that ‘page-write’. |
| profile\_id | Number | Not Null | Used to associate the ‘page-write’ to it’s original poster user profile ID. |
| note | Yes/No | Not Null | Determines whether the ‘page-write’ is a note or not a note. |
| happy | Yes/No | Not Null | Determines whether the feeling associated is born out of a happy emotion or a sad one. |

**Table Name**: tests

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | test\_id | | |
| **Foreign Key** | from, to | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| test\_id | AutoNumber | Not Null | Refers to each individual testimonial written between unique users. |
| from | Number | Not Null | Associated to the profile ID of the user who wrote the testimonial. |
| to | Number | Not Null | Associated to the profile ID of the user who the testimonial was written for. |
| rating | Number | Not Null | Rating that the ‘from’ user gives to the ‘to’ user. |
| content | Memo | Not Null | Content of the testimonial. |
| date\_written | Date/Time | Not Null | Date and time the testimonial was written at. |

**Table Name**: message\_threads

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | thread\_id | | |
| **Foreign Key** | from, to | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| thread\_id | AutoNumber | Not Null | Used to refer to when sorting messages in the ‘message’ page. (Helps knowing which message belong to which conversation.) |
| from | Number | Not Null | The participant A of the conversation. Stores the profile ID of that user. |
| to | Number | Not Null | The participant B of the conversation. Stores the profile ID of that user. |
| last\_updated\_on | Date/Time | Not Null | Date and time when last message was sent in this conversation. (Helps queue the conversation threads on the ‘message’ page in a regulated format.) |

**Table Name:** block\_list

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | <None> | | |
| **Foreign Key** | blocker, blockee | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| blocker | Number | Not Null | Stores the profile\_id of the person who has blocked. |
| blockee | Number | Not Null | Stores the profile\_id of the person who has been blocked. |

**Table Name**: comments

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | comment\_id | | |
| **Foreign Key** | poster, writes\_id\_c | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| comment\_id | AutoNumber | Not Null | Auto-generating comment IDs for individual unique comments. |
| poster | Number | Not Null | Stores the profile ID of the user that comments. |
| content\_c | Text | Not Null | Stores the content of the comment. |
| date\_written\_c | Date/Time | Not Null | Stores the date and time the user makes the comment. |
| writes\_id\_c | Number | Not Null | Refers to the write ID to which the comment belongs to. (Helps generating comments pertaining to a certain ’page-write’ easily.) |

**Table Name**: activity

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | activity\_id | | |
| **Foreign Key** | doer, doee, post | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| activity\_id | AutoNumber | Not Null | Auto-generating activity IDs for every unique activity that takes place on the website. |
| doer | Number | Not Null | Stores the profile ID of the user that does the acitvity. |
| type | Text | * started following * commented on * wrote a new * liked a * updated * published a | Refers to what type of activity is being done. |
| doee | Number | Not Null | Stores the profile ID of the user that the activity is being done on. |
| date\_time\_written | Date/Time | Not Null | Stores the date and time stamps of the activity taking place. |
| post | Number | Not Null | Stores the writes ID on which the activity is done. |

**Table Name**: req

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | req\_id | | |
| **Foreign Key** | from, to | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| req\_id | AutoNumber | Not Null | Auto-generating IDs for requests taking place over the website. |
| from | Number | Not Null | Stores the profile ID of the user that sends the ‘follow’ requests. |
| to | Number | Not Null | Stores the profile ID of the user that the ‘follow’ requests are sent to. |
| accepted | Yes/No | Not Null | Determines whether the request is accepted or not. (Helps displaying pending ‘follow requests.’) |
| date\_time | Date/Time | Not Null | Stamps date and time the request is sent on. |

**Table Name**: messages

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | message\_id | | |
| **Foreign Key** | thread\_id, sender | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| message\_id | AutoNumber | Not Null | Auto-genertaing IDs for all the messages being sent over the website. |
| content | Memo | Not Null | Content of the message sent by the user to another. |
| date\_written | Date/Time | Not Null | Date and time stamps of when the message was sent. |
| thread\_id | Number | Not Null | Refers to the ‘message\_threads’ table. (Help associating which message belongs to which conversation.) |
| sender | Number | Not Null | Stores the profile ID of the user that sends the message. (Helps website-formatting for conversation-style message display.) |

**Table Name**: session\_logs

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | session\_no | | |
| **Foreign Key** | profile\_id | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| session\_no | AutoNumber | Not Null | Auto-genertaing IDs for all the taking place over the website. |
| profile\_id | Number | Not Null | Refers to the profile IDs of the users that login. |
| ip\_id | Text | Not Null | Stores the fetched value of the IP Address that a user’s device is allocated. |
| date\_time\_login | Date/Time | Not Null | Date and time stamps of when the login takes place. |
| brow\_name | Text | Not Null | Stores the fetched browser name that the user uses to access the website. |
| brow\_ver | Text | Not Null | Stores the fetched browser name that the user uses to access the website. |

**Table Name**: music\_data, movies\_data, books\_data, food\_data, tv\_data, team\_data, places\_data.

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | \*\_d\_id | | |
| **Foreign Key** | <None> | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| \*\_d\_id | AutoNumber | Not Null | Auto-generating IDs for all the favorite-inputs taking place over the website. |
| \*\_name | Text | Not Null | Stores the data related to the ID of the favorites that are filled. |

**Table Name**: music\_junc, movies\_junc, books\_junc, food\_junc, tv\_junc, team\_junc, places\_junc.

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | <None> | | |
| **Foreign Key** | profile\_id, \*\_d\_id | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| profile\_id | Number | Not Null | Stores the profile IDs of the user that is associated with the other field in the table.  (Refers to the data in the above table.) |
| \*\_d\_id | Number | Not Null | Stores the data related to the favorite of the IDs belonging to the above table. |

**Table Name**: int

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | int\_id | | |
| **Foreign Key** | <None> | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| int\_id | AutoNumber | Not Null | Auto-generating IDs for every entry made into the ‘interests’ tab. |
| int\_name | Text | Not Null | The data associated with the interest IDs is stored here. |

**Table Name**: int\_junc

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | <None> | | |
| **Foreign Key** | int\_id, profile\_id | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| int\_id | Number | Not Null | Refers to the above table and stores the value of the primary key here.  (To help form a junction between which profile relates to which fields.) |
| profile\_id | Number | Not Null | Stores the profile IDs here which relate to the other field. (Again- forming a junction.) |

**Table Name**: int\_fav

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | int\_fav\_id | | |
| **Foreign Key** | <None> | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| int\_fav\_id | AutoNumber | Not Null | Auto-generating IDs for every entry made into the ‘favorites’ tab. |
| int\_fav | Text | Not Null | The data associated with the favorites IDs is stored here. |

**Table Name**: int\_fav\_junc

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | <None> | | |
| **Foreign Key** | int\_fav\_id, profile\_id | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| int\_fav\_id | Number | Not Null | Refers to the above table and stores the value of the primary key here.  (To help form a junction between which profile relates to which fields.) |
| profile\_id | Number | Not Null | Stores the profile IDs here which relate to the other field. (Again- forming a junction.) |

**Table Name:** picture\_library

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | photo\_id | | |
| **Foreign Key** | user\_path | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| photo\_id | AutoNumber | Not Null | Refers to the unique photo ID the user uploads. |
| user\_path | Number | Not Null | Refers to the user’s folder where the photos are stored. |
| folder\_type\_path | Text | Trips/Parties/Events/You | Refers to the type of photo the user uploaded so that it is displayed in that category only. |
| pic\_name | Text | Not Null | Refers to the name of the picture that the photo is saved as client-side. |
| add\_date | Date/Time | Not Null | Used to order photos in the photo-library. |
| caption | Text | Not Null | Stores the caption for that specific photo. |

**Table Name:** livesessions

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | session\_id | | |
| **Foreign Key** | host\_id | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| session\_id | AutoNumber | Not Null | Refers to the session the user hosted. |
| sub\_name | Text | Not Null | Stores the name of the subject of the session. |
| host\_id | Number | Not Null | Refers the profile\_id of the person hosting the session. |
| start\_time | Date/Time | Not Null | Notes the session start time. |
| end\_time | Date/Time | Not Null | Notes the session end time. |
| content | Text | Not Null | Stores the content of that specific session. |
| attachment\_path | Text | None | Stores the path of the attached file the session may have. |

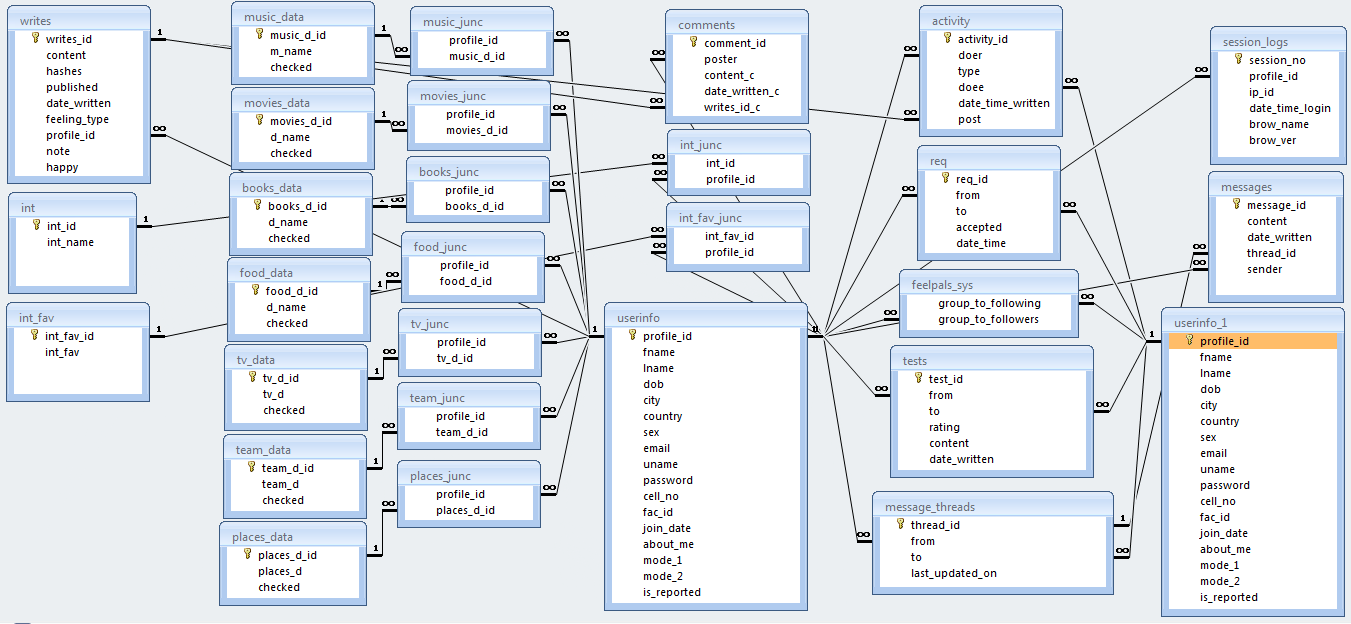
**Table Name:** bit\_data

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | bits\_id | | |
| **Foreign Key** | session\_id (livesessions) | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| bits\_id | AutoNumber | Not Null | Refers to the unique bit the user sends. |
| bit | Text | Not Null | Stores the content of the bit. |
| sender | Number | Not Null | Refers the profile\_id of the person sending the bit. |
| session\_id | Number | Not Null | Notes the session ID to which the bit belongs to. |
| bit\_date\_written | Date/Time | Not Null | Notes the time bit was sent. |

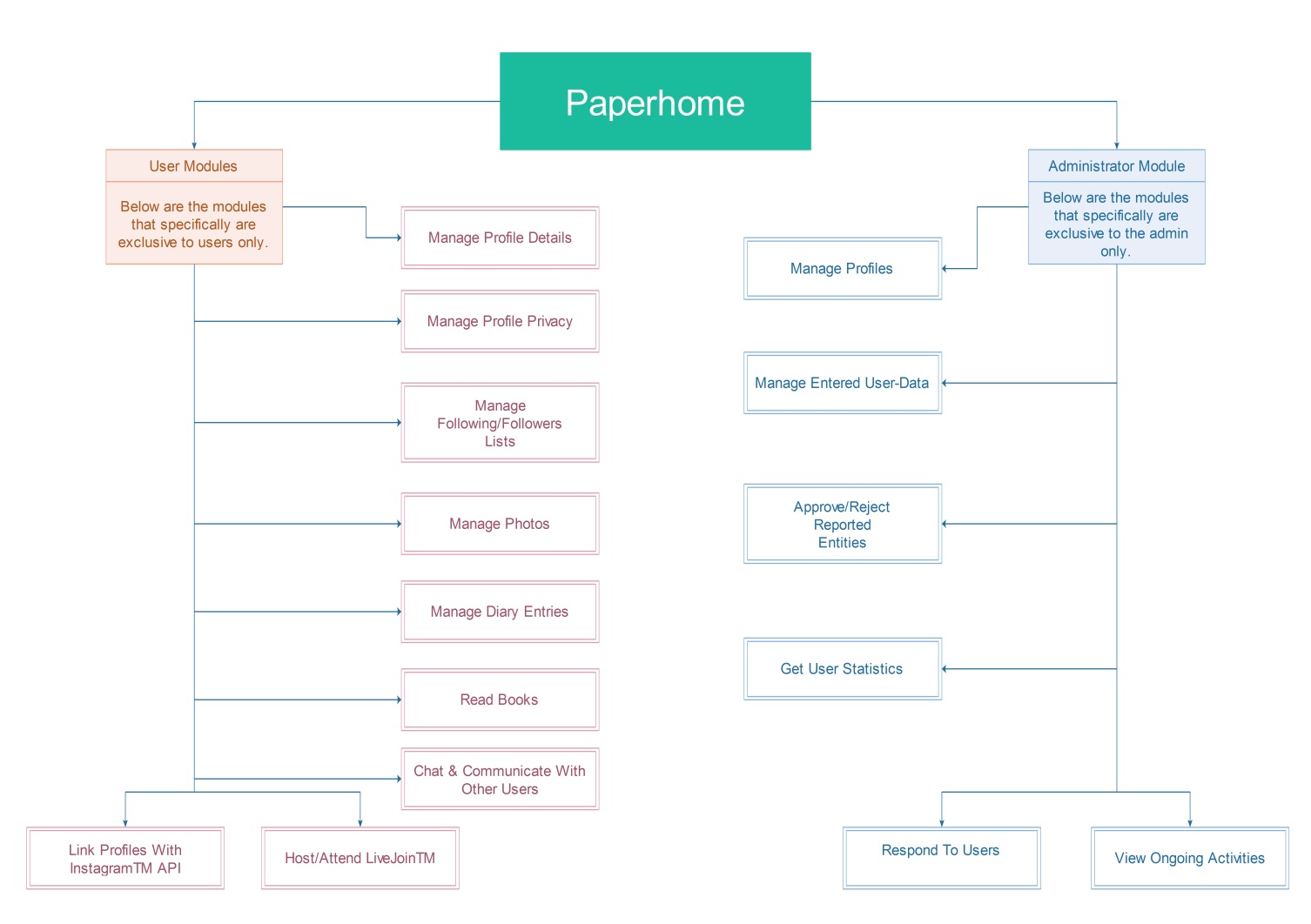
**Table Name:** library\_data

|  |  |  |  |
| --- | --- | --- | --- |
| **Primary Key** | book\_id | | |
| **Foreign Key** | None | | |
|  | | | |
| **Field Name** | **DataType** | **Constraints/Lookup** | **Description** |
| book\_id | AutoNumber | Not Null | Refers to the book stored. |
| book\_name | Text | Not Null | Stores the name of the book. |
| file\_path | Number | Not Null | Refers the file path the book is stored in. |

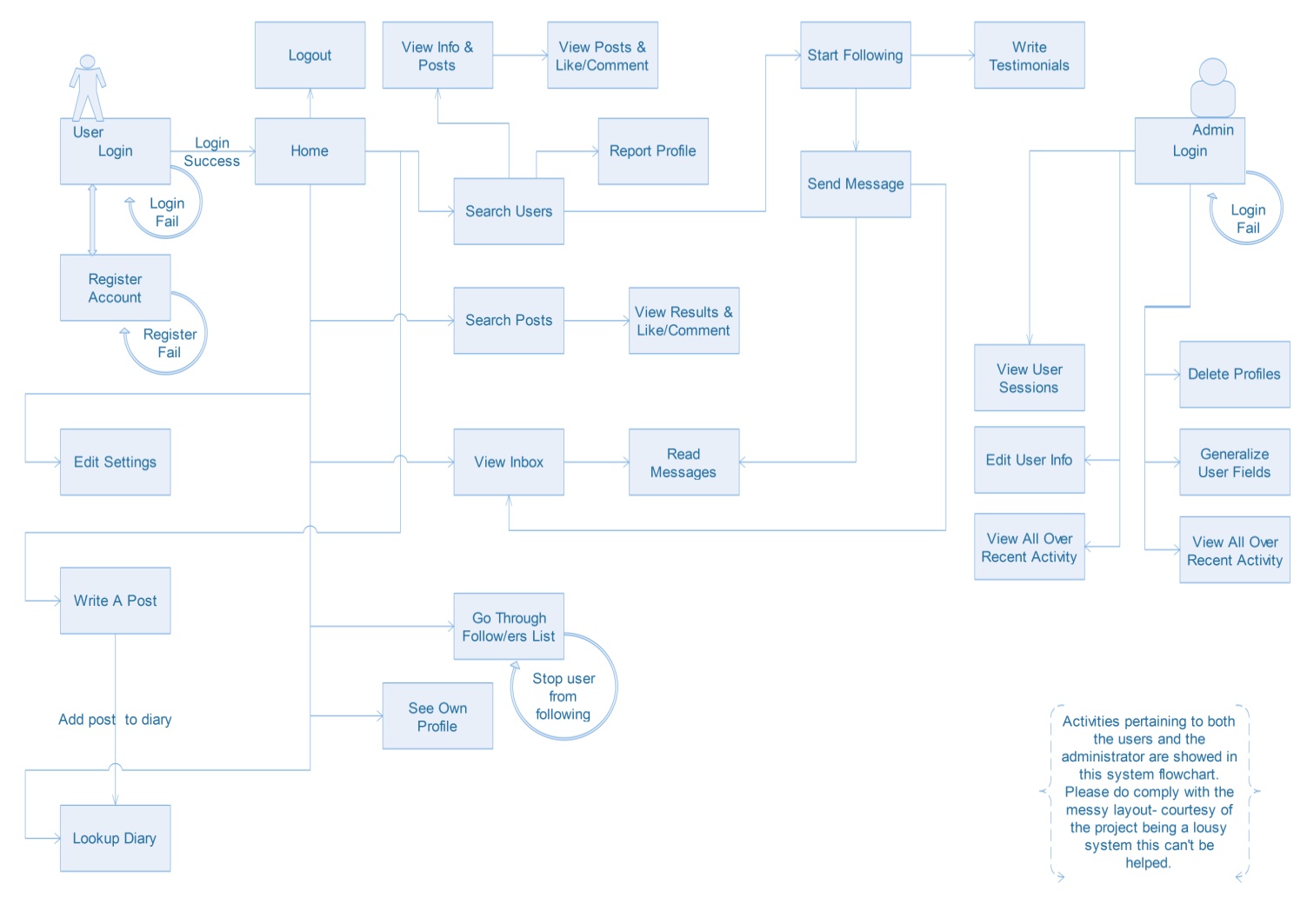
**Database Design**



**System Architecture**

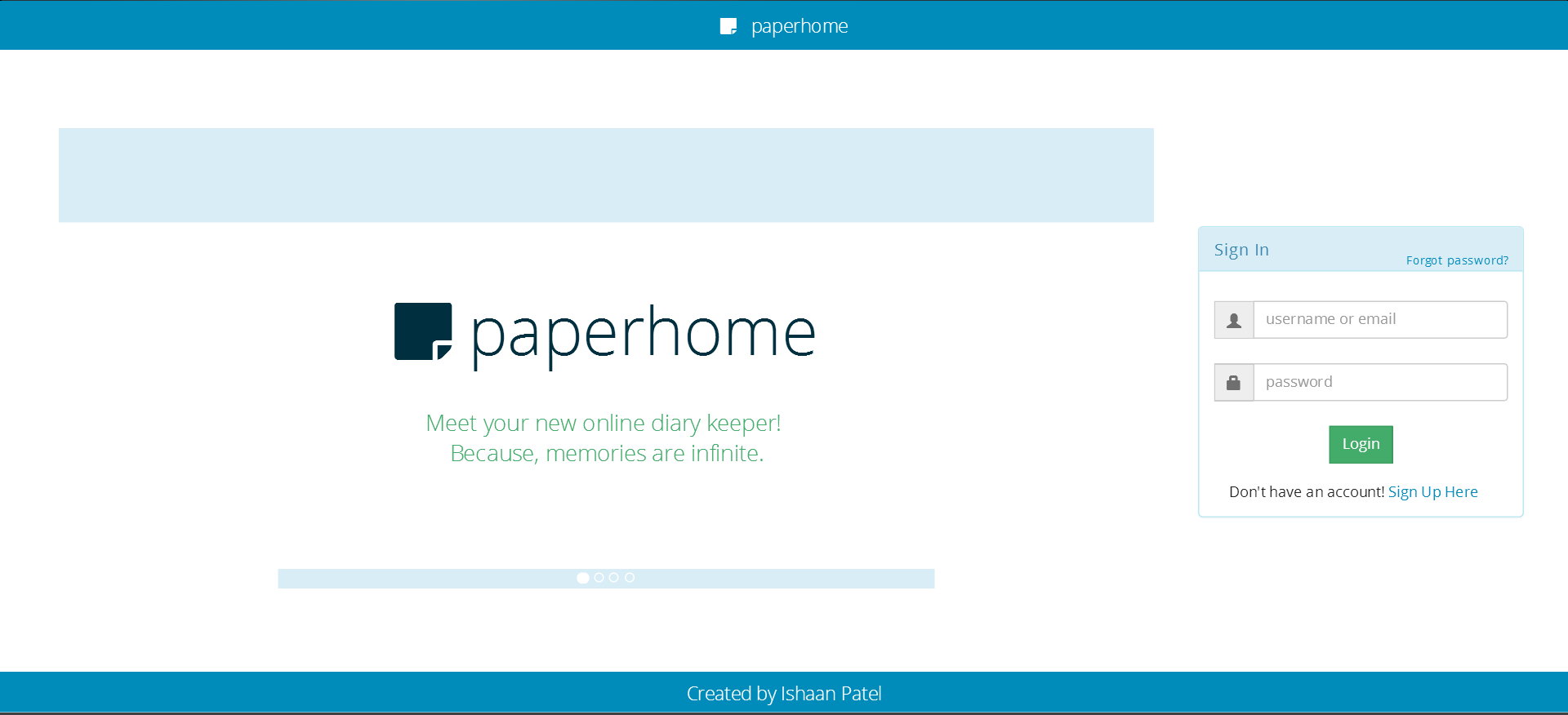


**System Flowchart**

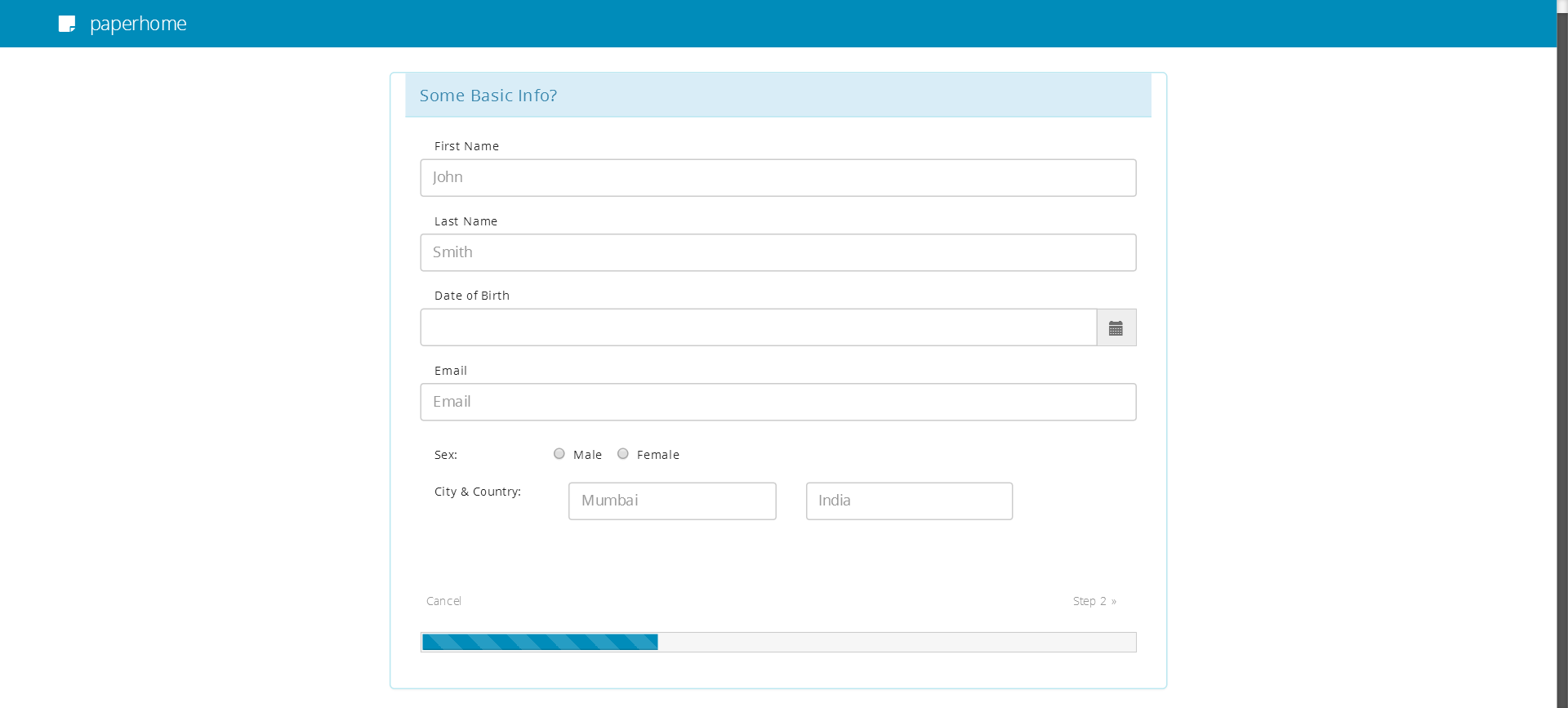


**Form & Report Design**

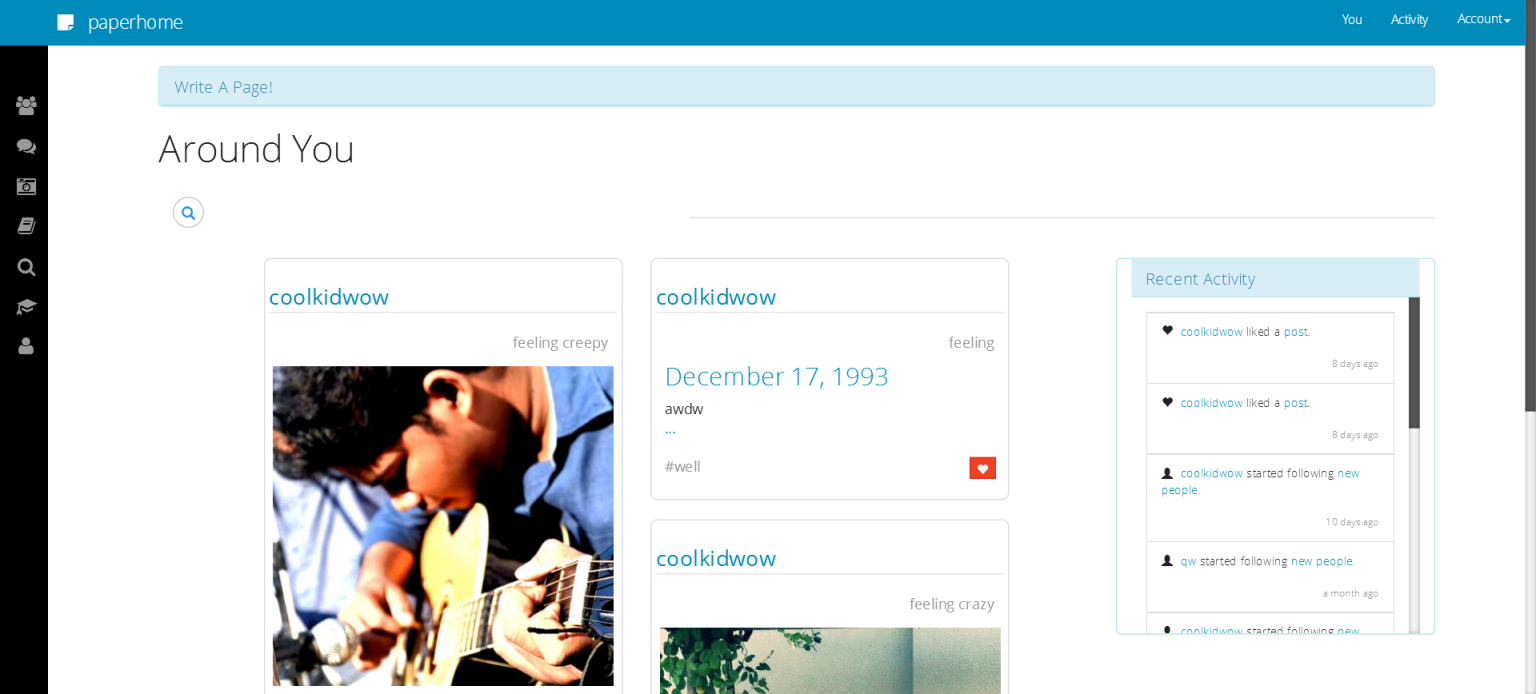
**Login**

****

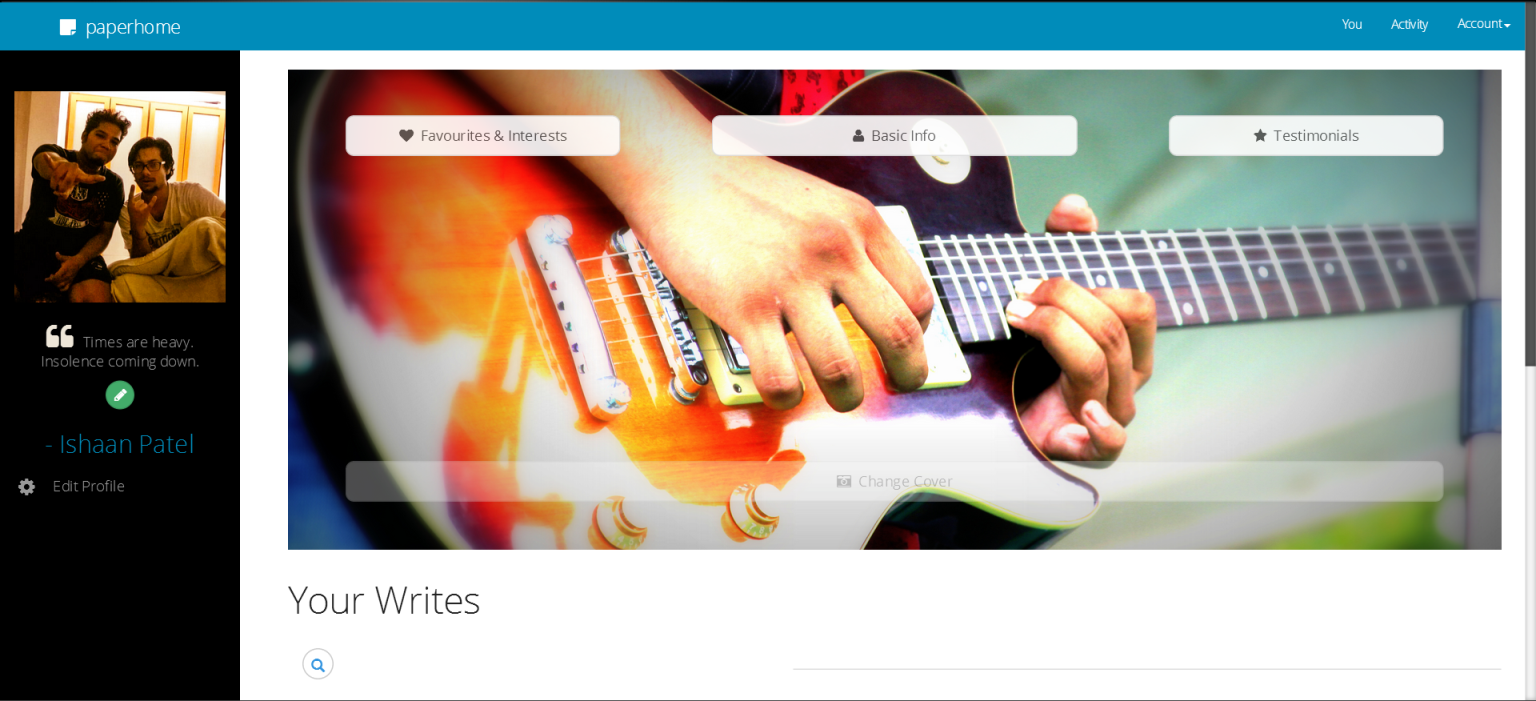
**Registeration**

****

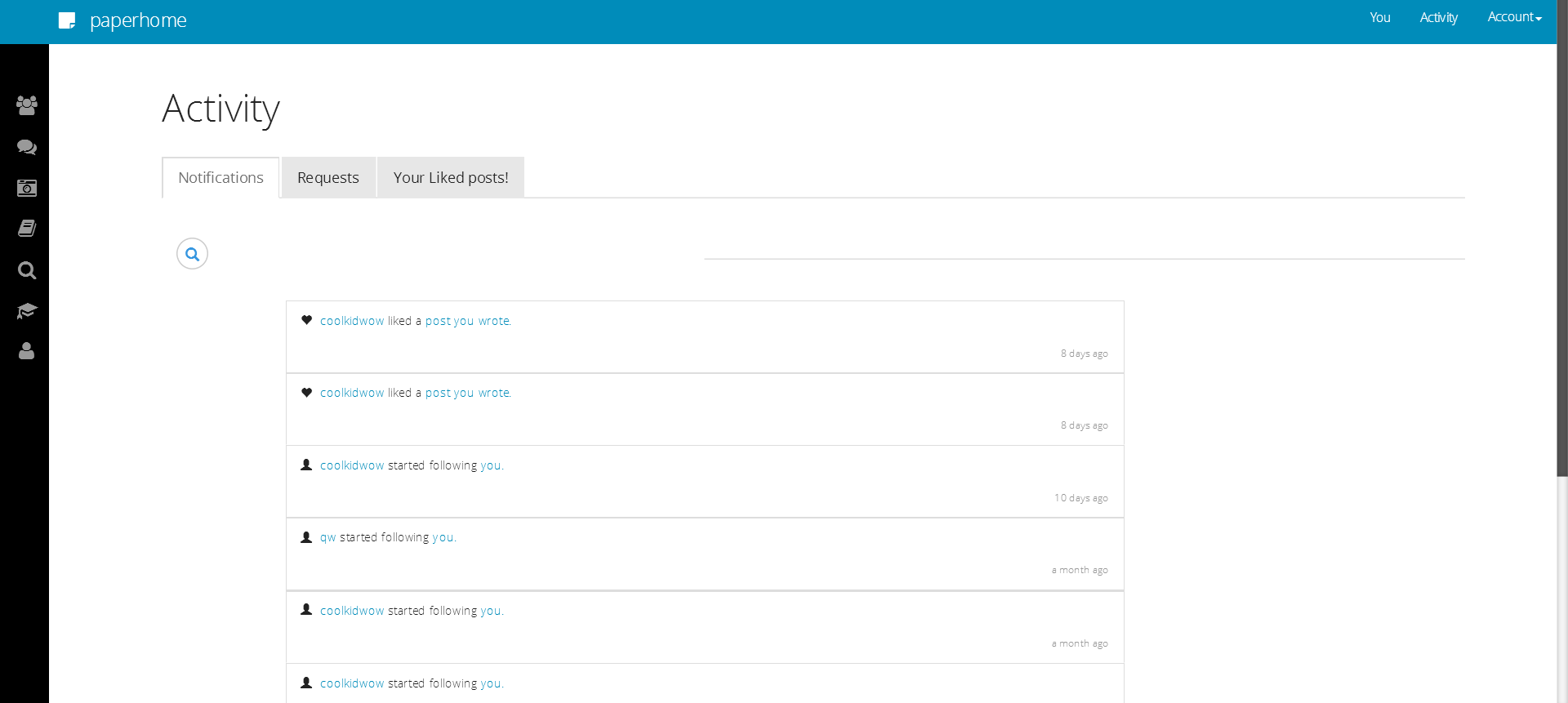
**Home**

****

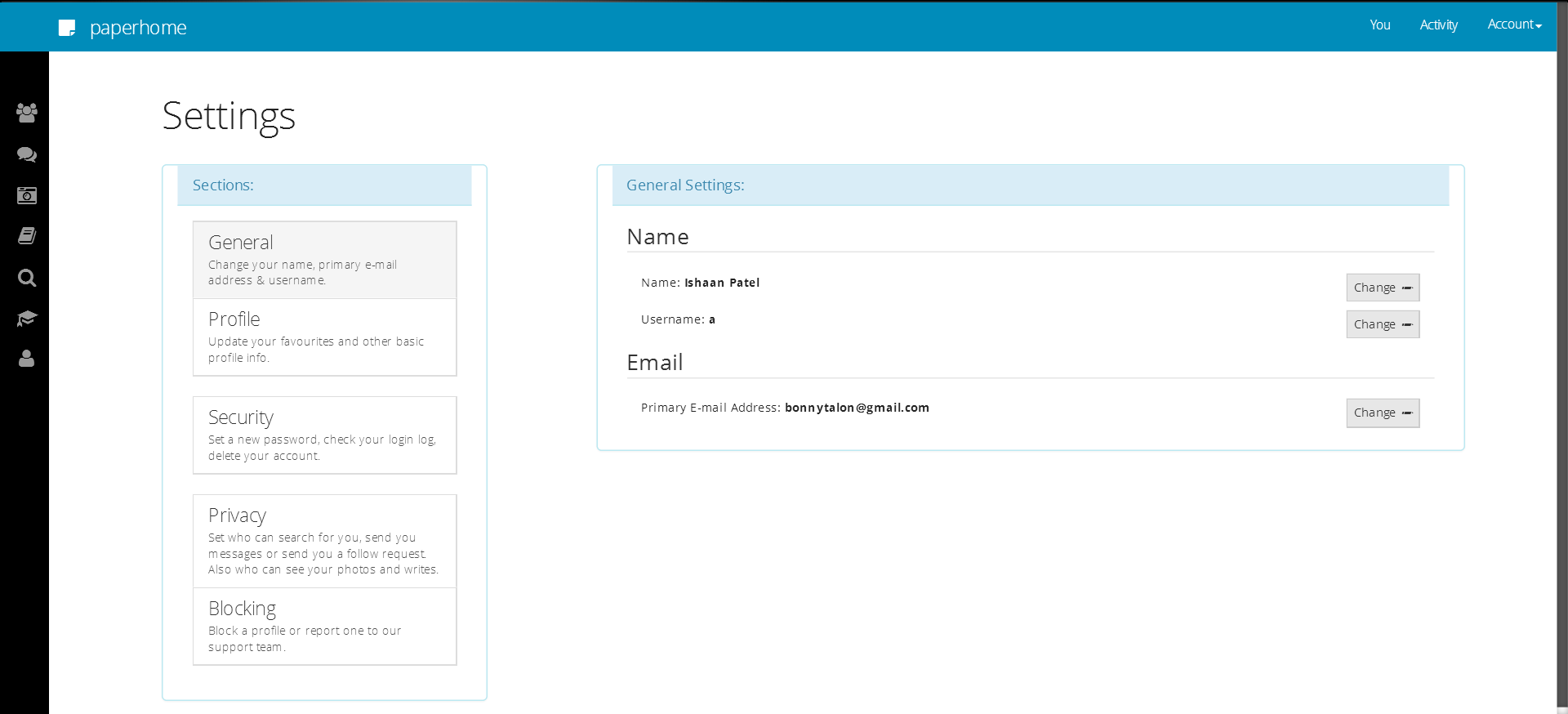
**Your Profile**

****

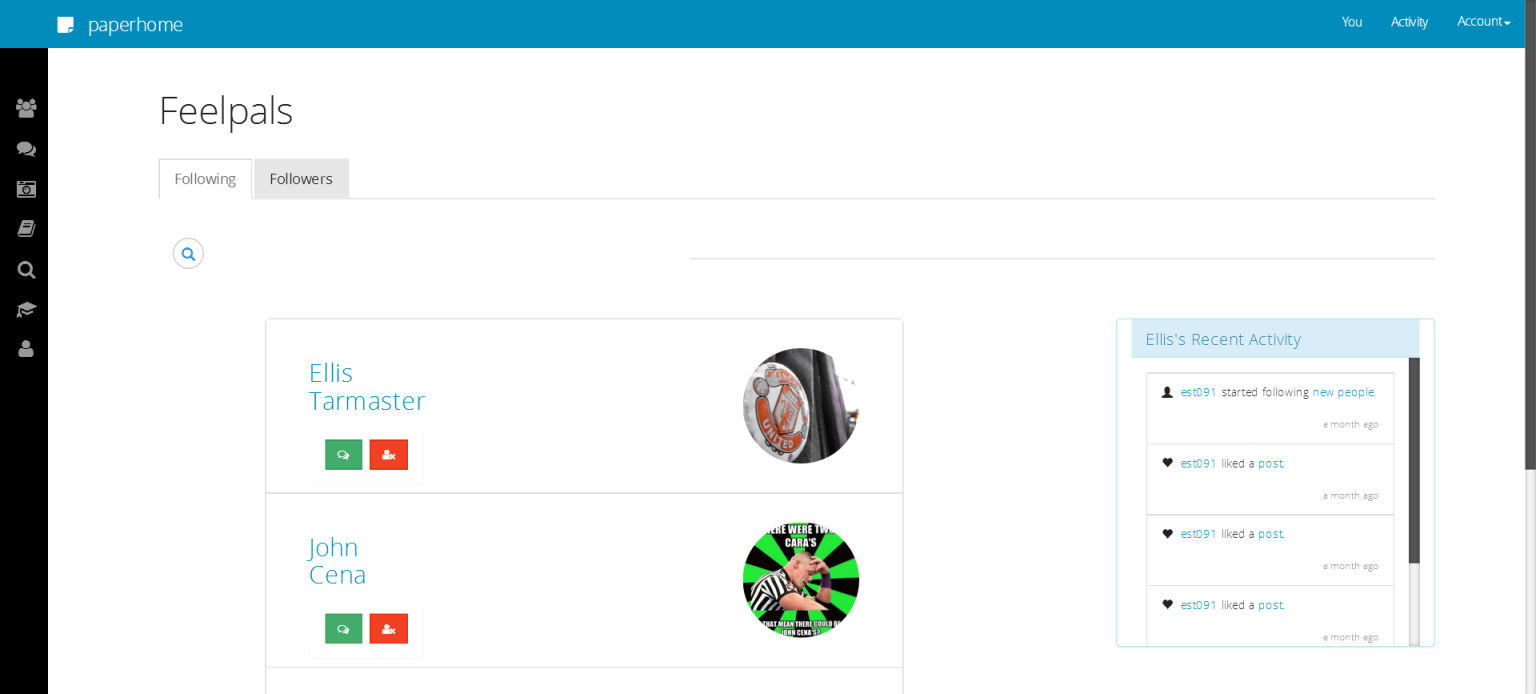
**Activity**

****

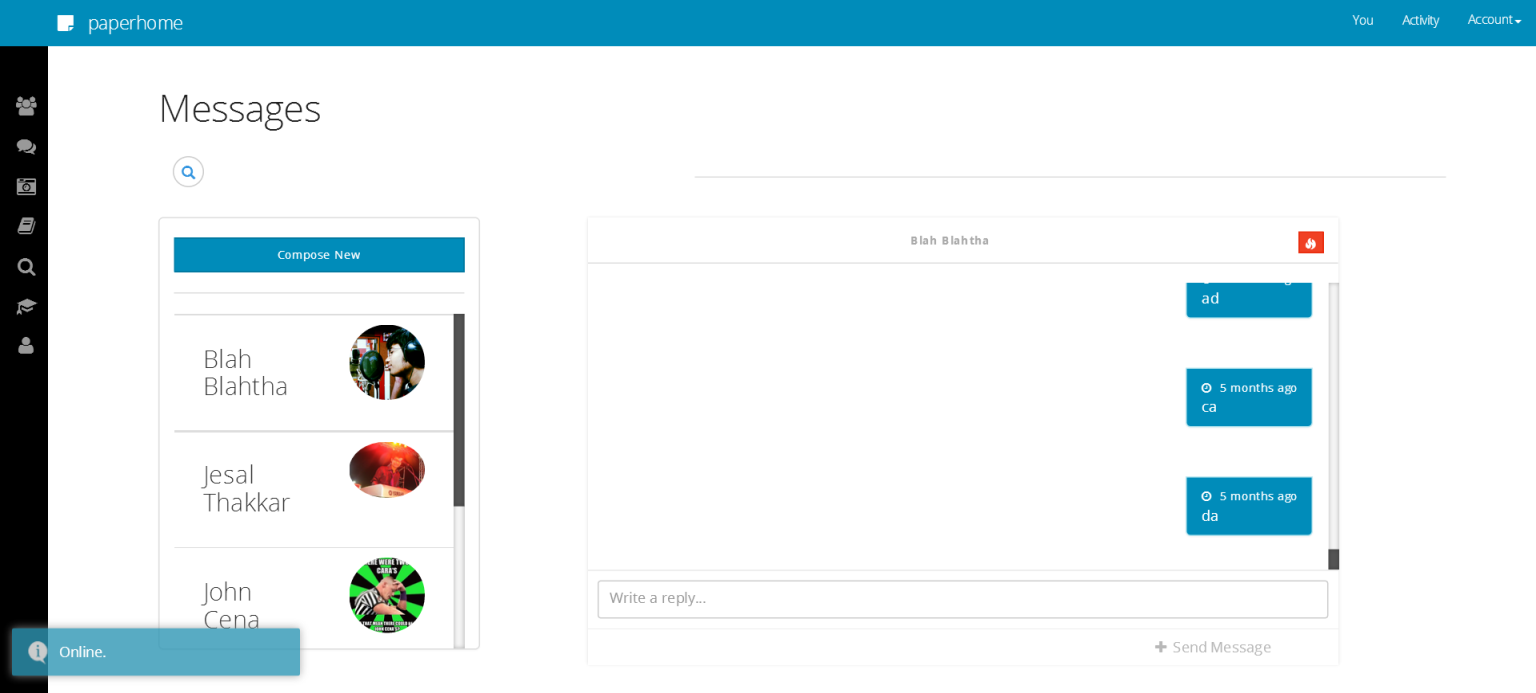
**Settings**

****

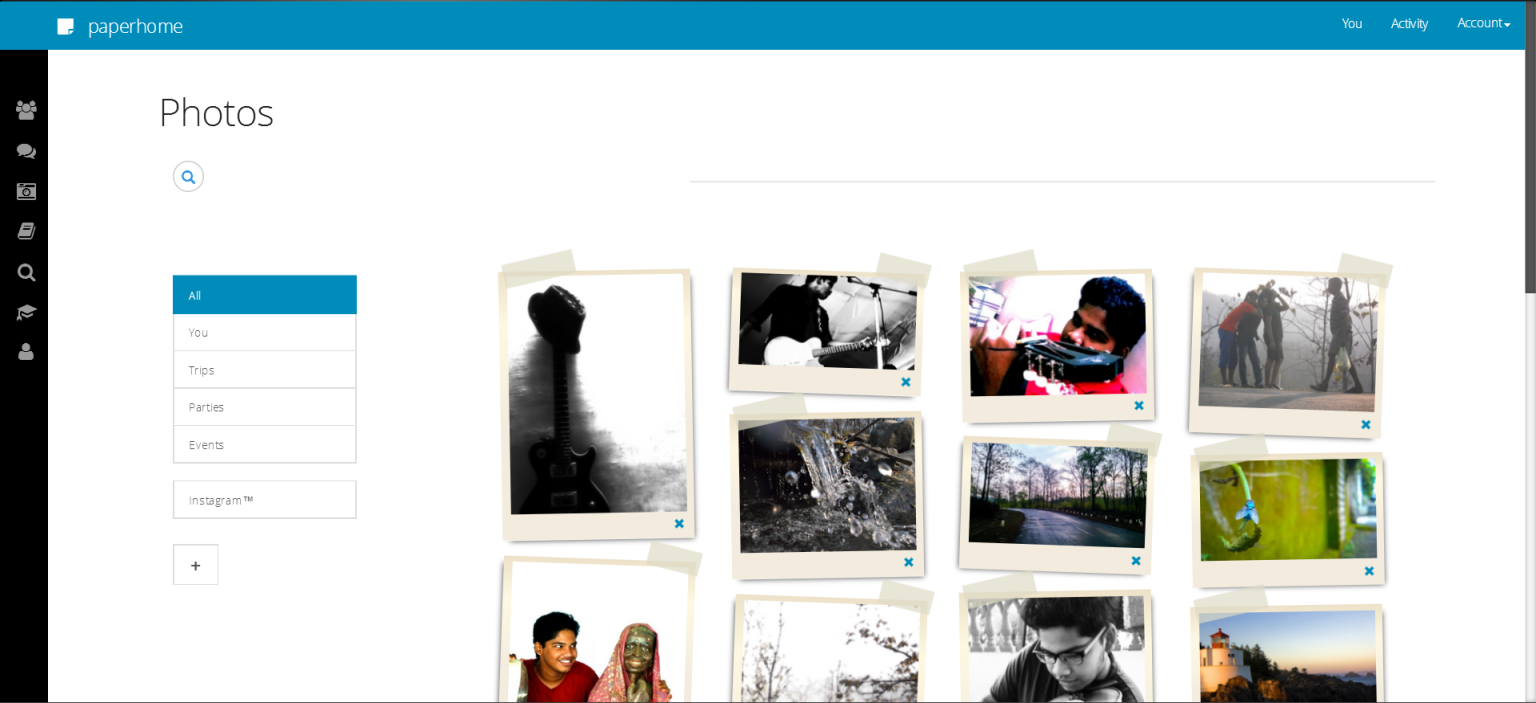
**Feelpals**

****

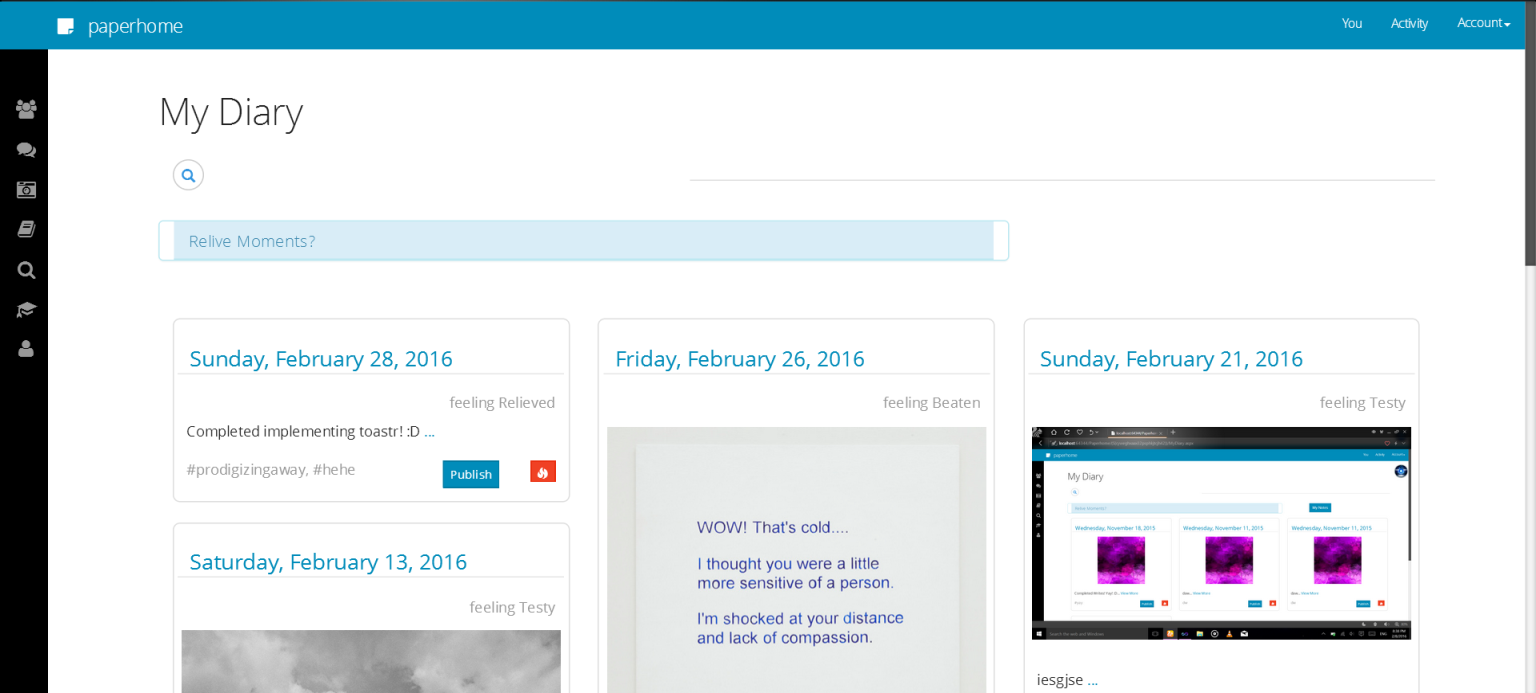
**Messages**

****

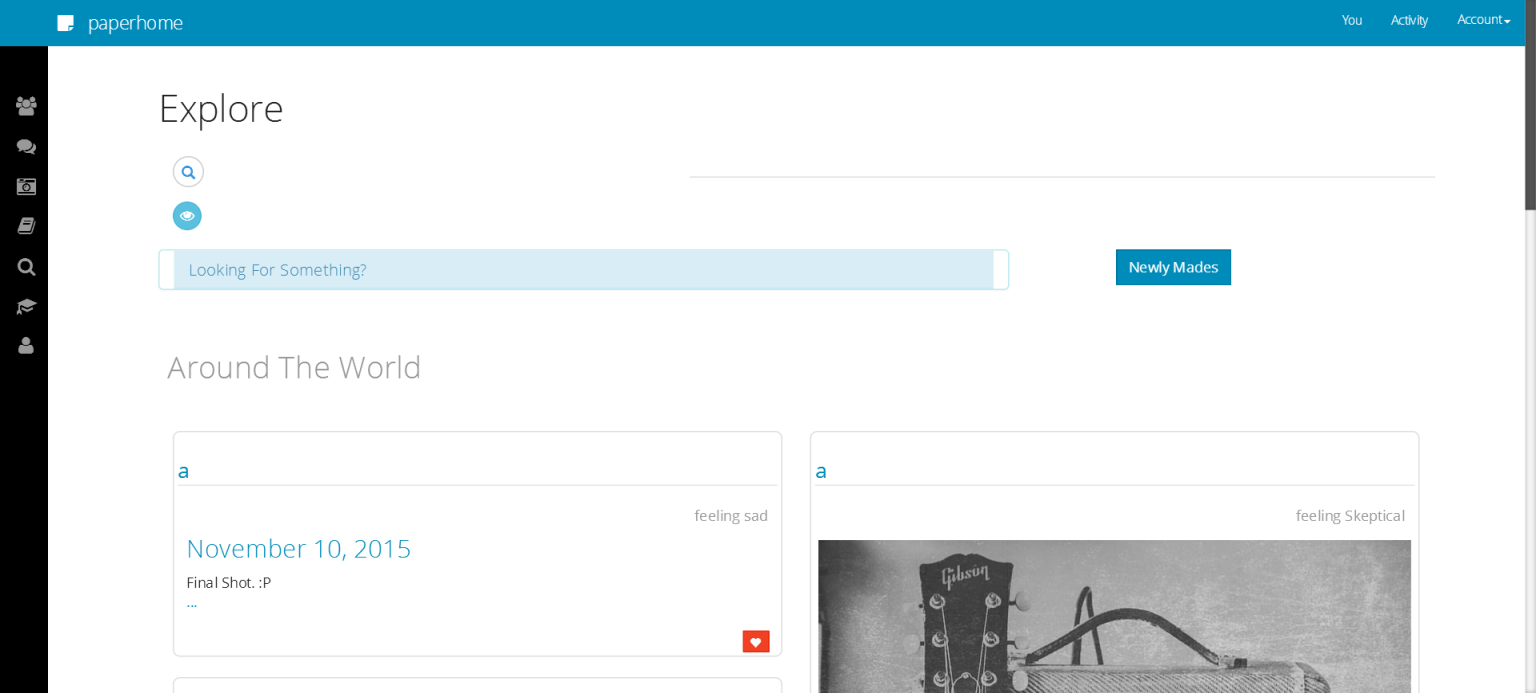
**Photos**

****

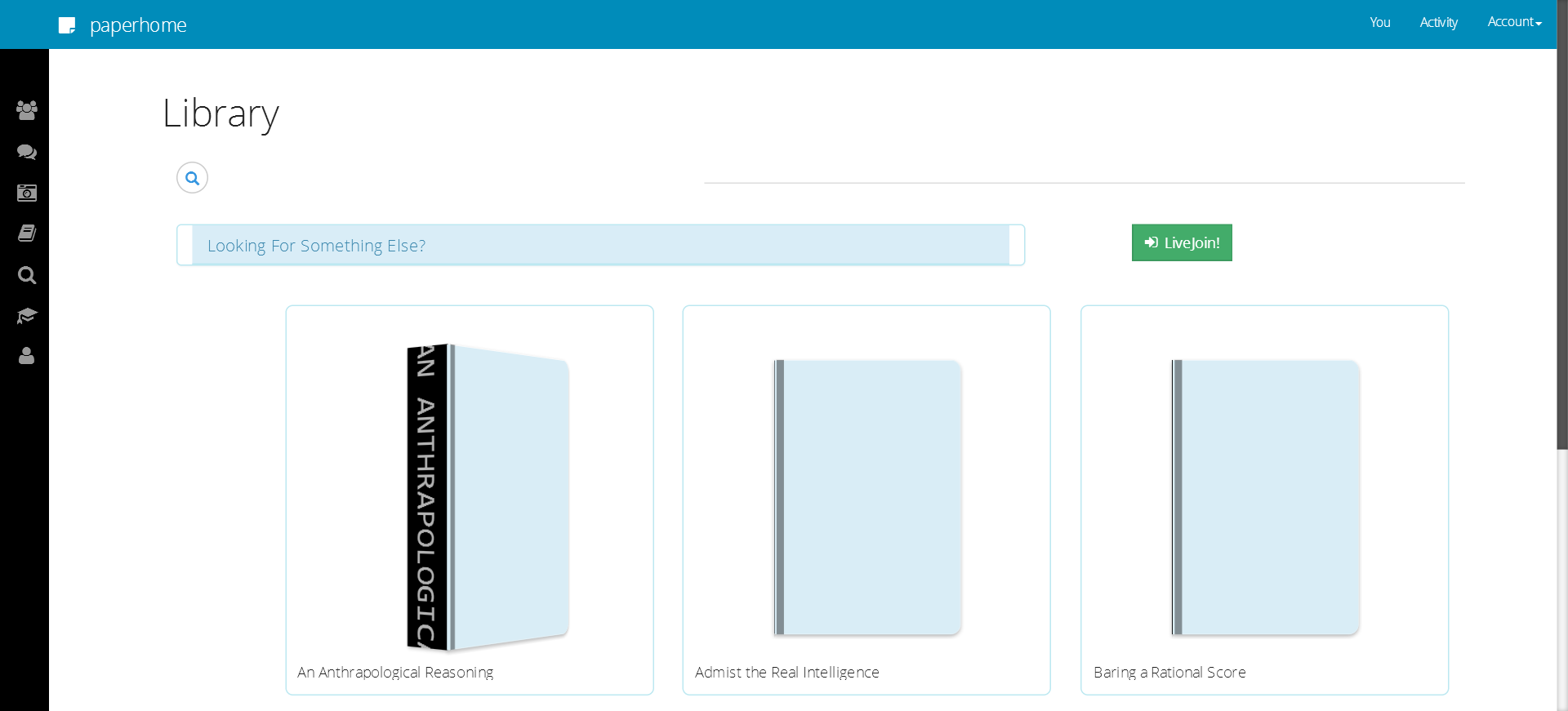
**Your Diary**

****

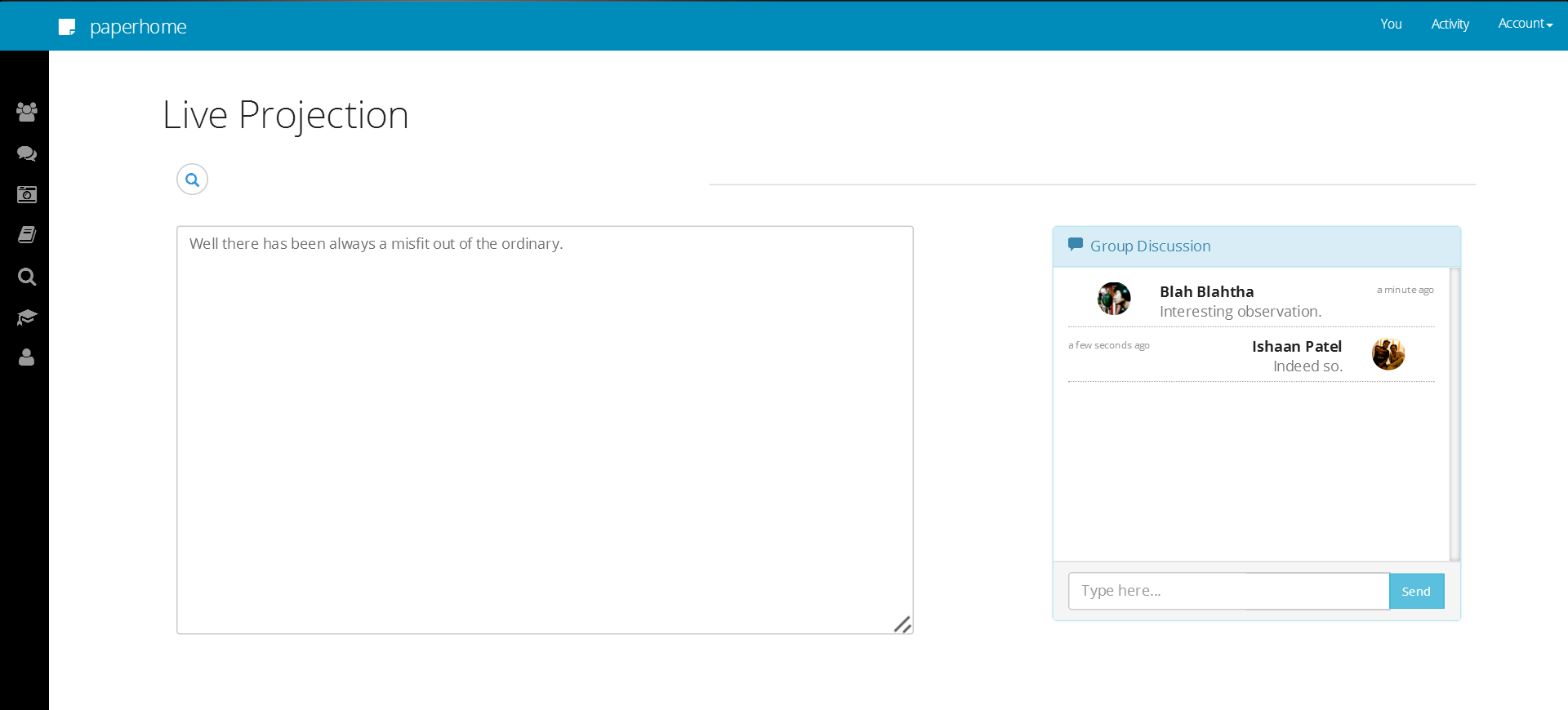
**Explore**

****

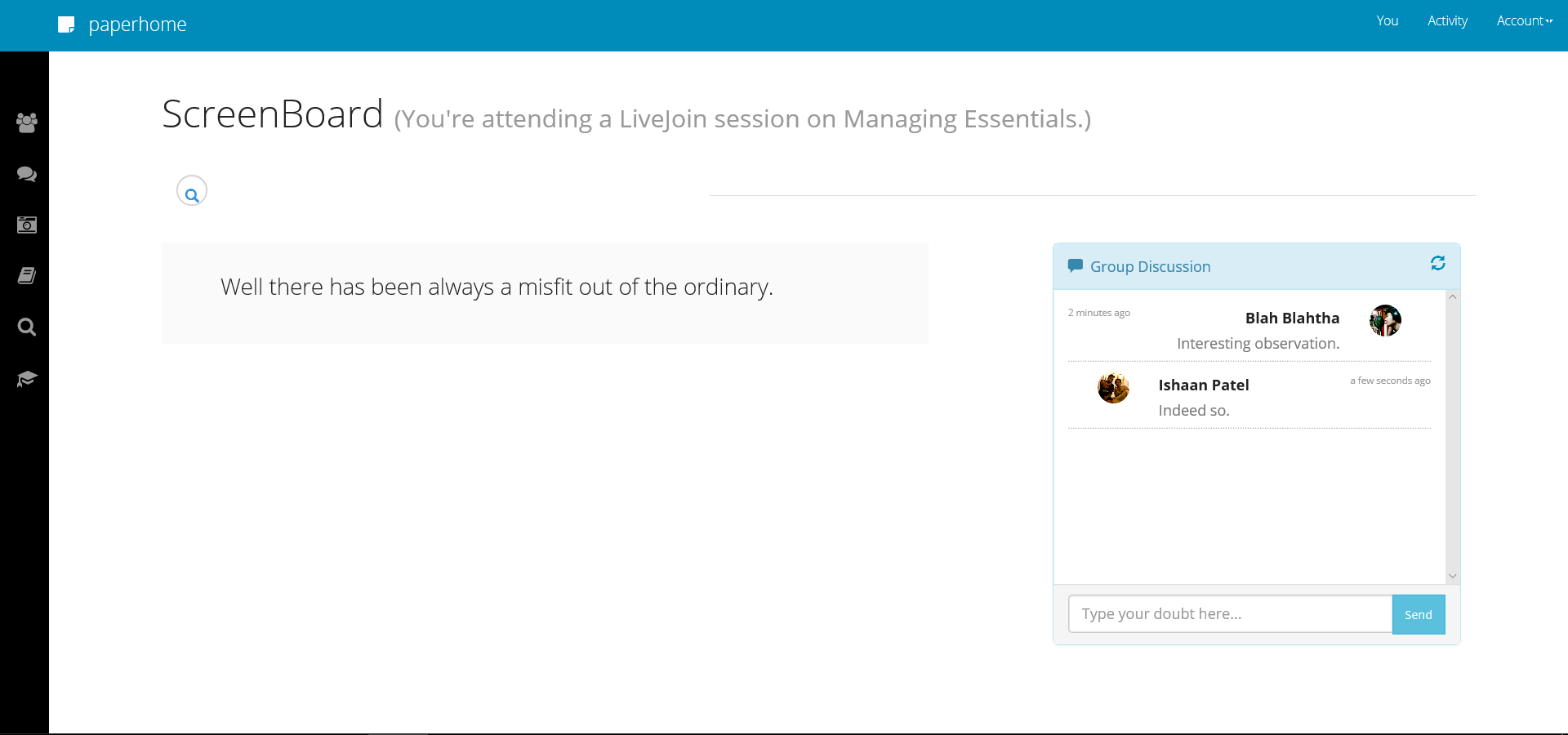
**Library**

****

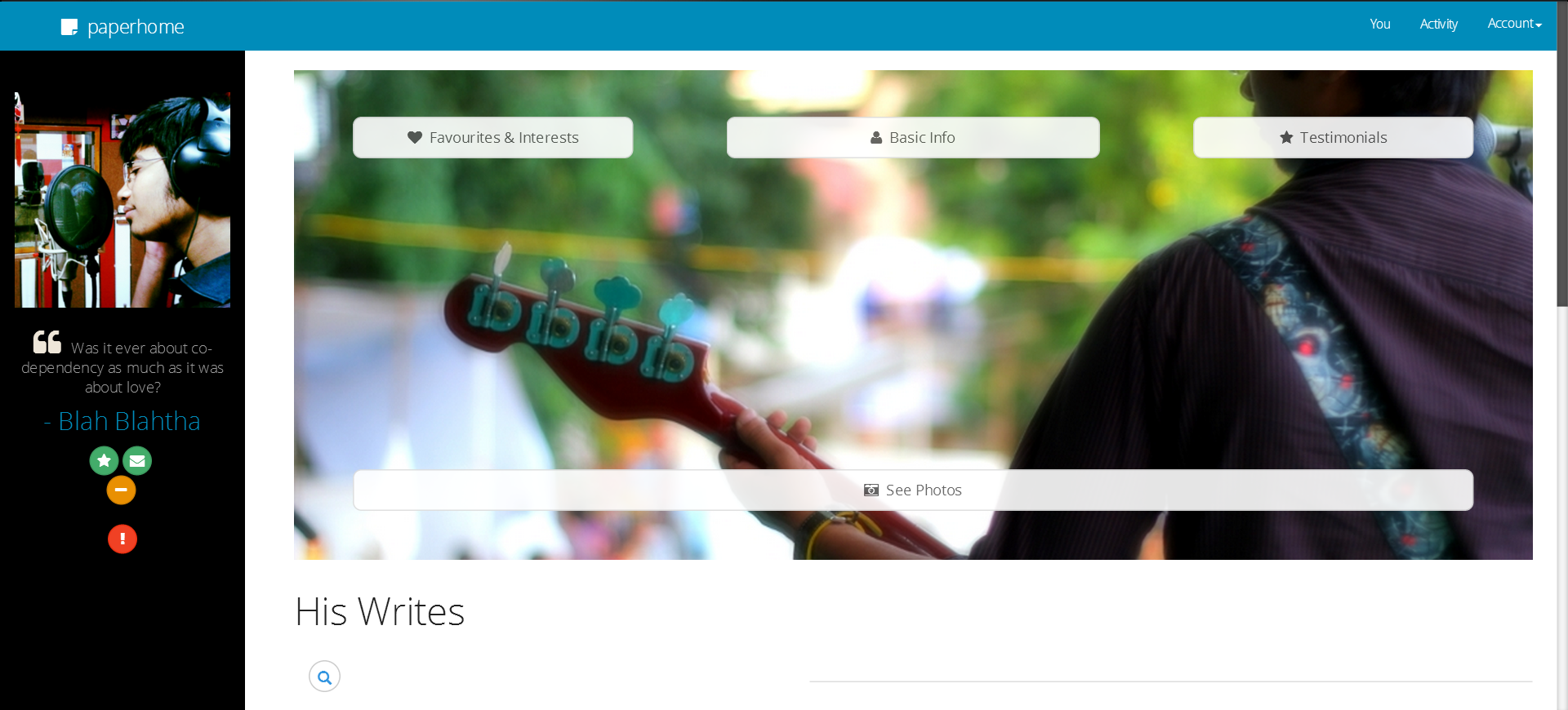
**LiveJoinTM Host Session**

****

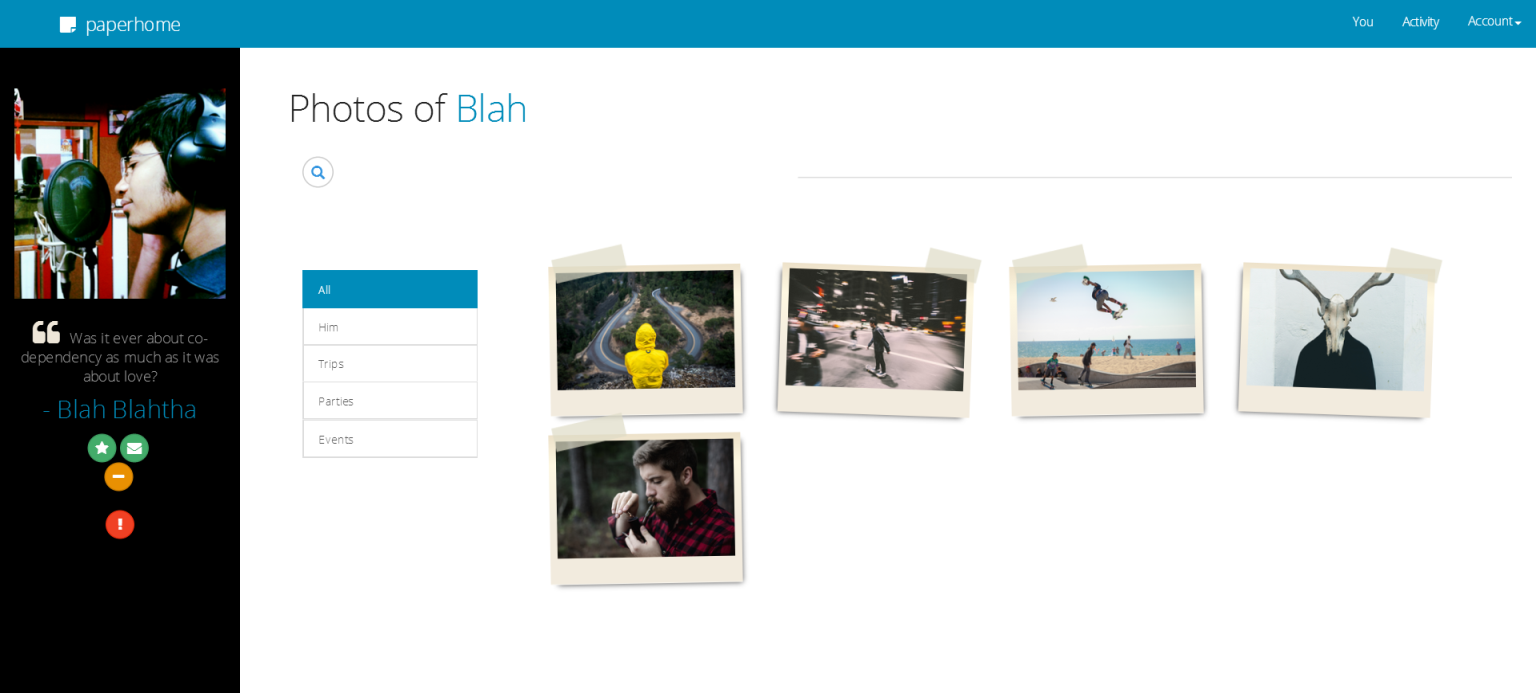
**LiveJoinTM Attending Session**

****

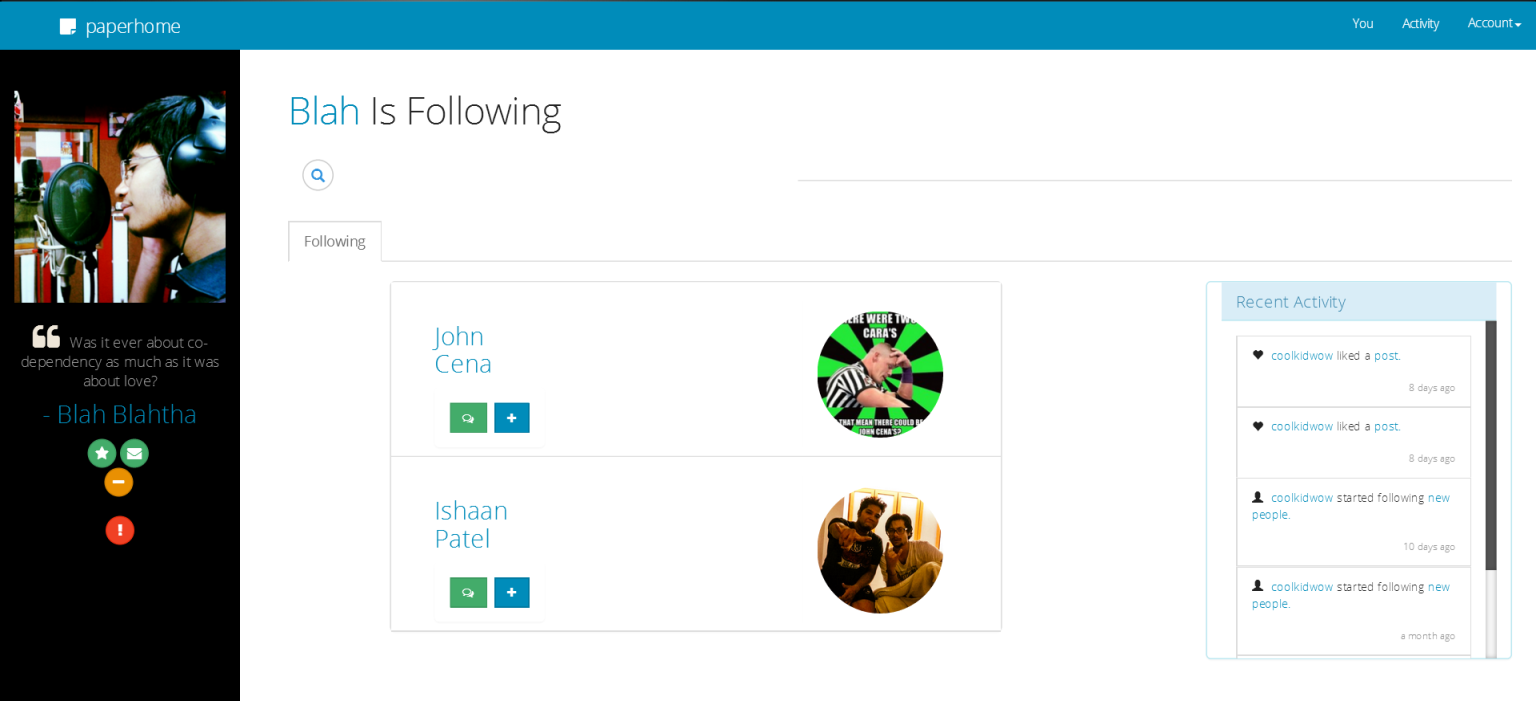
**Your Friend’s Profile (Openbook & Writes)**

****

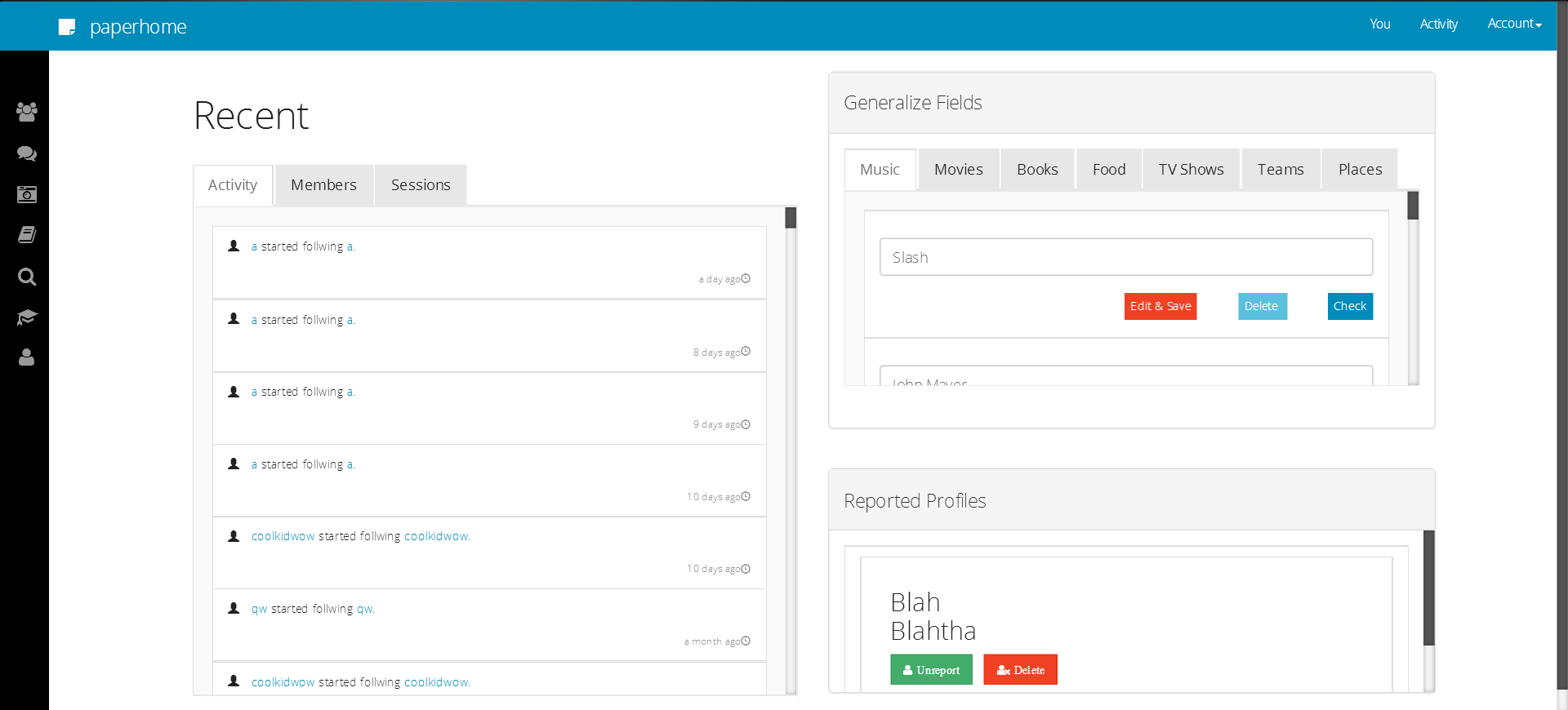
**Your Friend’s Profile (Photos)**

****

**Your Friend’s Profile (People They Follow)**

****

**Main Admin Interface**

****

**Coding**

**Coding Approach/Style**

**Requirement:**

* As a time-table creation website, most of the needs were focused on flow of data from each wizard step.
* Javascript OLEs were used to store data in a project-requisitive format and forwarded to multiple servlets that processed these into a succinct time-table out-put format.
* These servlet-outputs were then post-processed by assigning a self-constructed CSS class that made it possible to display the jargon of raw, fresh information into a beautiful time-table like structure.

**Design:**

* It was necessary that the time-table creation be as smooth and easy as possible, hence a wizard was the first choice, since it took inputs from users at a very reasonable rate and provided guidance/validations along the way so that the user isn’t bombarded with unnecessarily high-levels of UI complexity.
* Although it was very much possible to just display the raw output the algorithm provided directly, it made more sense to make that output be in a classified and beautiful CSS format so that it acutally resembled a time-table like structure. This was a true test of both my skills at javascript, analytical abilities, CSS and all my reasoning capabilities as well.

**Index for Code Snippets:**

1. Login (with IP fetching)
2. Like Post
3. Unlike Post
4. Comment Post
5. Redirect To Profile
6. Accept Request
7. Decline Request
8. Delete Account
9. Stop a User From Following You
10. Logout
11. Stop Following a User
12. Start a New Conversation (Includes message sending)
13. Send Request
14. Report Profile
15. Delete Post
16. Write Testimonial
17. Delete Conversation
18. Publish Post
19. Happy Diary Search
20. Sad Diary Search
21. Upload Photos WebHandler
22. AI- Part Scheme
23. Search Feelpals
24. Search Posts

**Code Snippets**

Login (with IP fetching):

Protected Sub loginsub(ByVal Sender As Object, ByVal e As EventArgs)

con.Open()

cmd = New OleDbCommand("SELECT ([password]) FROM userinfo WHERE ([uname]= '" & uname\_log.Value & "')", con)

dr = cmd.ExecuteReader

If (dr.Read) Then

If (dr(0).ToString = pass\_log.Value) Then

cmd = New OleDbCommand("SELECT [profile\_id] FROM userinfo WHERE ([uname]= '" & uname\_log.Value & "')", con)

dr = cmd.ExecuteReader

If (dr.Read) Then

Session("idsess") = dr("profile\_id")

Response.Write(Session("idsess"))

Else

Response.Write("dasda")

End If

Dim IPAdd As String = String.Empty

IPAdd = Request.ServerVariables("HTTP\_X\_FORWARDED\_FOR")

Dim mysess As String

mysess = Convert.ToString(Session("idsess"))

Dim intDay As Date

intDay = Date.Now

Dim objBrwInfo As HttpBrowserCapabilities = Request.Browser

Dim browname As String

browname = objBrwInfo.Browser

Dim browver As String

browver = objBrwInfo.Version

If String.IsNullOrEmpty(IPAdd) Then

IPAdd = Request.ServerVariables("REMOTE\_ADDR")

cmdiprem = New OleDbCommand("INSERT INTO session\_logs(profile\_id, ip\_id, date\_time\_login, brow\_name, brow\_ver) VALUES('" & mysess & "', '" & IPAdd & "', '" & intDay & "', '" & browname & "', '" & browver & "')", con)

cmdiprem.Connection = con

cmdiprem.ExecuteNonQuery()

Else

cmdip = New OleDbCommand("INSERT INTO session\_logs(profile\_id, ip\_id, brow\_name, brow\_ver) VALUES('" & mysess & "', '" & IPAdd & "', '" & intDay & "', '" & browname & "', '" & browver & "')", con)

cmdip.Connection = con

cmdip.ExecuteNonQuery()

End If

Response.Redirect("Home.aspx")

Else

Response.Write("Wrong Authentification.")

End If

Else

Response.Write("Sign up instead?")

End If

con.Close()

End Sub

Like Post:

Protected Sub like\_post(ByVal Sender As Object, ByVal e As CommandEventArgs)

Dim commandArgsAccept As String = e.CommandArgument.ToString()

Dim writeid As String = commandArgsAccept.ToString

Session("tempsess") = writeid

Dim wrisess As String

wrisess = Convert.ToInt32(Session("tempsess"))

Dim mysess As String

mysess = Convert.ToString(Session("idsess"))

Dim intDay As Date

intDay = Date.Now

con.Open()

cmd = New OleDbCommand("SELECT \* FROM activity WHERE ([doer]=" & mysess & " AND [type]='liked a' AND [post]=" & wrisess & ")", con)

cmd.Connection = con

dr = cmd.ExecuteReader

If (dr.Read) Then

Response.Write("You've already liked this post.")

Else

cmd = New OleDbCommand("INSERT INTO activity(doer, type, post, date\_time\_written) VALUES('" & mysess & "','liked a', '" & wrisess & "', '" & intDay & "')", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

End If

End Sub

Unlike Post:

Protected Sub unlike\_post(ByVal Sender As Object, ByVal e As CommandEventArgs)

Dim commandArgsAccept As Integer = e.CommandArgument

Dim writeid As Integer = commandArgsAccept

Session("tempsess") = writeid

Dim wrisess As Integer

wrisess = Convert.ToInt32(Session("tempsess"))

Dim mysess As Integer

mysess = Convert.ToInt32(Session("idsess"))

con.Open()

cmd = New OleDbCommand("DELETE \* FROM activity WHERE [doer] = " & mysess & " AND [type] = 'liked a' AND [post] = " & wrisess & "", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Response.Redirect("Activity.aspx")

End Sub

Comment Post:

Protected Sub comment\_post(ByVal Sender As Object, ByVal e As CommandEventArgs)

Session.Remove("tempsess2")

Dim commandArgsAccept As Integer = e.CommandArgument

Dim proid As Integer = commandArgsAccept

Session("tempsess2") = proid

Dim wrisess As Integer

wrisess = Convert.ToInt32(Session("tempsess2"))

Dim mysess As Integer

mysess = Convert.ToInt32(Session("idsess"))

Dim intDay As Date

intDay = Date.Now

con.Open()

Dim txtBox As HtmlInputText

Dim messtxta As String

For Each rItem As RepeaterItem In Repeater1.Items

txtBox = DirectCast(rItem.FindControl("comment\_txt"), HtmlInputText)

If Not IsNothing(txtBox) Then

If txtBox.Value.Length > 0 Then

messtxta = txtBox.Value

End If

End If

Next

cmd = New OleDbCommand("INSERT INTO comments(poster, content\_c, date\_written\_c, Writes\_id\_c) VALUES('" & mysess & "','" & messtxta.Replace("'", "''") & "', '" & intDay & "', '" & wrisess & "')", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

'activity input'

con.Open()

cmd = New OleDbCommand("INSERT INTO activity(doer, type, date\_time\_written, post) VALUES('" & mysess & "', 'commented on a', '" & intDay & "', '" & wrisess & "') ", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Me.DataBind()

End Sub

Redirect To Profile:

Sub red(ByVal Sender As Object, ByVal e As CommandEventArgs)

Dim commandArgsAccept As String = e.CommandArgument.ToString()

Dim value1 As String = commandArgsAccept.ToString

Session("prosess") = value1

If (Session("prosess") = Session("idsess")) Then

Response.Redirect("MyProfile.aspx")

Else

Response.Redirect("XsProfile.aspx?id=" & value1)

End If

End Sub

Accept Request:

Protected Sub accept(ByVal Sender As Object, ByVal e As CommandEventArgs)

Dim commandArgsAccept As Integer = e.CommandArgument

Dim hisid As Integer = commandArgsAccept

Session("tempsess") = hisid

Dim hissess As Integer

hissess = Convert.ToInt32(Session("tempsess"))

Dim mysess As Integer

mysess = Convert.ToInt32(Session("idsess"))

Dim intDay As Date

intDay = Date.Now

con.Open()

cmd = New OleDbCommand("UPDATE req SET [accepted]= True WHERE ([from] = " & hissess & " AND [to] = " & mysess & ")", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

cmd = New OleDbCommand("INSERT INTO feelpals\_sys(group\_to\_following, group\_to\_followers) VALUES(" & hissess & ", " & mysess & ")", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

'activity input'

con.Open()

cmd = New OleDbCommand("INSERT INTO activity(doer, type, doee, date\_time\_written) VALUES(" & hissess & ", 'started following', " & mysess & ", '" & intDay & "') ", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Response.Redirect("Activity.aspx")

End Sub

Decline Request:

Protected Sub decline(ByVal Sender As Object, ByVal e As CommandEventArgs)

Dim commandArgsAccept As Integer = e.CommandArgument

Dim hisid As Integer = commandArgsAccept

Session("tempsess") = hisid

Dim hissess As Integer

hissess = Convert.ToInt32(Session("tempsess"))

Dim mysess As Integer

mysess = Convert.ToInt32(Session("idsess"))

con.Open()

cmd = New OleDbCommand("DELETE FROM req WHERE ([from] = " & hissess & " AND [to] = " & mysess & ")", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Response.Redirect("Activity.aspx")

End Sub

Delete Account:

Protected Sub delacc(ByVal Sender As Object, ByVal e As EventArgs)

Dim mysess As Integer

mysess = Convert.ToInt32(Session("idsess"))

con.Open()

cmd = New OleDbCommand("SELECT ([password]) FROM userinfo WHERE ([profile\_id]= " & mysess & ")", con)

dr = cmd.ExecuteReader

If (dr.Read) Then

Session("pass") = dr(0).ToString()

If (Session("pass") = wrepw.Value) Then

con.Close()

con.Open()

cmd = New OleDbCommand("DELETE \* FROM userinfo WHERE (profile\_id=" & mysess & ")", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Response.Redirect("MainPage.aspx")

Else

Response.Write("Enter correct password.")

End If

End If

End Sub

Stop User From Following You:

Sub stopf(ByVal Sender As Object, ByVal e As CommandEventArgs)

Dim commandArgsAccept As Integer = e.CommandArgument

Dim proid As Integer = commandArgsAccept

Session("stopf") = proid

Dim prosess As Integer

prosess = Convert.ToInt32(Session("stopf"))

Dim mysess As Integer

mysess = Convert.ToInt32(Session("idsess"))

con.Open()

cmd = New OleDbCommand("DELETE \* FROM feelpals\_sys WHERE ([group\_to\_followers] = " & mysess & " AND [group\_to\_following] = " & prosess & ")", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Response.Redirect("Feelpals.aspx")

End Sub

Logout:

Protected Sub logoutsess(ByVal Sender As Object, ByVal e As EventArgs)

Session.Clear()

Response.Redirect("MainPage.aspx")

End Sub

Stop Following a User:

Sub unf(ByVal Sender As Object, ByVal e As CommandEventArgs)

Dim commandArgsAccept As Integer = e.CommandArgument

Dim proid As Integer = commandArgsAccept

Session("unf") = proid

Dim prosess As Integer

prosess = Convert.ToInt32(Session("unf"))

Dim mysess As Integer

mysess = Convert.ToInt32(Session("idsess"))

con.Open()

cmd = New OleDbCommand("DELETE \* FROM feelpals\_sys WHERE ([group\_to\_following] = " & mysess & " AND [group\_to\_followers] = " & prosess & ")", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Response.Redirect("Feelpals.aspx")

End Sub

Start a New Conversation:

Protected Sub sendmess(ByVal Sender As Object, ByVal e As CommandEventArgs)

Session.Remove("tempsess")

Dim commandArgsAccept As Integer = e.CommandArgument

Dim proid As Integer = commandArgsAccept

Session("tempsess") = proid

Dim prosess As Integer

prosess = Convert.ToInt32(Session("tempsess"))

Dim mysess As Integer

mysess = Convert.ToInt32(Session("idsess"))

Dim intDay As Date

intDay = Date.Now

con.Open()

cmd = New OleDbCommand("SELECT [thread\_id] FROM message\_threads WHERE (([from] = " & prosess & " AND [to] = " & mysess & ") OR ([to] = " & prosess & " AND [from] = " & mysess & "))", con)

cmd.Connection = con

dr = cmd.ExecuteReader

If (dr.Read) Then

Dim thread As Integer

Dim txtBox As HtmlInputText

Dim messtxta As String

thread = Convert.ToInt32(dr(0).ToString())

For Each rItem As RepeaterItem In Repeater1.Items

txtBox = DirectCast(rItem.FindControl("messtxt"), HtmlInputText)

If Not IsNothing(txtBox) Then

If txtBox.Value.Length > 0 Then

messtxta = txtBox.Value

End If

End If

Next

cmd2 = New OleDbCommand("INSERT INTO messages(content, sender, thread\_id, date\_written) VALUES('" & messtxta.Replace("'", "''") & "', '" & mysess & "', '" & thread & "', '" & intDay & "')", con)

cmd2.Connection = con

cmd2.ExecuteNonQuery()

cmd = New OleDbCommand("UPDATE message\_threads SET [last\_updated\_on] = '" & intDay & "' WHERE ([thread\_id] = " & thread & ")", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

Response.Redirect("Feelpals.aspx")

Else

con.Close()

con.Open()

cmd3 = New OleDbCommand("INSERT INTO message\_threads([from], [to]) VALUES('" & mysess & "', '" & prosess & "')", con)

cmd3.Connection = con

cmd3.ExecuteNonQuery()

cmd4 = New OleDbCommand("SELECT [thread\_id] FROM message\_threads WHERE (([from] = " & prosess & " AND [to] = " & mysess & ") OR ([to] = " & prosess & " AND [from] = " & mysess & "))", con)

dr = cmd4.ExecuteReader

If (dr.Read) Then

Dim newthread As Integer

newthread = Convert.ToInt32(dr(0).ToString())

Dim txtBox As HtmlInputText

Dim messtxta As String

For Each rItem As RepeaterItem In Repeater1.Items

txtBox = DirectCast(rItem.FindControl("messtxt"), HtmlInputText)

If Not IsNothing(txtBox) Then

If txtBox.Value.Length > 0 Then

messtxta = txtBox.Value

End If

End If

Next

cmd5 = New OleDbCommand("INSERT INTO messages(content, sender, thread\_id, date\_written) VALUES('" & messtxta.Replace("'", "''") & "', '" & mysess & "', '" & newthread & "', '" & intDay & "')", con)

cmd5.Connection = con

cmd5.ExecuteNonQuery()

cmd = New OleDbCommand("UPDATE message\_threads SET [last\_updated\_on] = '" & intDay & "' WHERE ([thread\_id] = " & newthread & ")", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

Response.Redirect("Feelpals.aspx")

End If

End If

con.Close()

End Sub

Send Request:

Sub f(ByVal Sender As Object, ByVal e As CommandEventArgs)

Dim commandArgsAccept As Integer = e.CommandArgument

Dim proid As Integer = commandArgsAccept

Session("f") = proid

Dim prosess As Integer

prosess = Convert.ToInt32(Session("f"))

Dim mysess As Integer

mysess = Convert.ToInt32(Session("idsess"))

Dim intDay As Date

intDay = Date.Now

con.Open()

cmd = New OleDbCommand("INSERT INTO req([from], [to], date\_time) VALUES(" & mysess & "," & prosess & ",'" & intDay & "')", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Response.Redirect("XsProfile.aspx")

End Sub

Report Profile:

Sub repprosub(ByVal Sender As Object, ByVal e As CommandEventArgs)

Dim commandArgsAccept As Integer = e.CommandArgument

Dim prosess As Integer

prosess = Convert.ToInt32(e.CommandArgument)

con.Open()

cmd = New OleDbCommand("UPDATE userinfo SET [is\_reported] = True WHERE ([profile\_id] = " & prosess & ")", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Response.Redirect("XsProfile.aspx")

End Sub

Delete Post:

Protected Sub burnwrite(ByVal Sender As Object, ByVal e As CommandEventArgs)

Dim commandArgsAccept As String = e.CommandArgument.ToString()

Dim writeid As String = commandArgsAccept.ToString

Session("tempsess") = writeid

Dim wrisess As String

wrisess = Convert.ToInt32(Session("tempsess"))

con.Open()

cmd = New OleDbCommand("DELETE \* FROM writes WHERE [writes\_id] = " & wrisess & "", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Response.Redirect("MyDiary.aspx")

End Sub

Write Testimonial:

Sub wrtest(ByVal Sender As Object, ByVal e As CommandEventArgs)

Dim commandArgsAccept As Integer = e.CommandArgument

Dim proid As Integer = commandArgsAccept

Session("unf") = proid

Dim prosess As Integer

prosess = Convert.ToInt32(Session("unf"))

Dim mysess As Integer

mysess = Convert.ToInt32(Session("idsess"))

Dim intDay As Date

intDay = Date.Now

con.Open()

Dim txtBox As HtmlInputText

Dim messtxtaw As String

For Each rItem As RepeaterItem In Repeater12.Items

txtBox = DirectCast(rItem.FindControl("Text1"), HtmlInputText)

If Not IsNothing(txtBox) Then

If txtBox.Value.Length > 0 Then

messtxtaw = txtBox.Value

End If

End If

Next

Dim rated As Integer

For Each rItem As RepeaterItem In Repeater12.Items

txtBox = DirectCast(rItem.FindControl("rate"), HtmlInputText)

If Not IsNothing(txtBox) Then

If txtBox.Value.Length > 0 Then

rated = txtBox.Value

End If

End If

Next

cmd = New OleDbCommand("INSERT INTO tests([to], [from], content, rating, date\_written) VALUES(" & prosess & ", " & mysess & ", '" & messtxtaw.Replace("'", "''") & "', '" & rated & "', '" & intDay & "')", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Response.Redirect("XsProfile.aspx")

End Sub

Delete Conversation:

Sub del\_con(ByVal Sender As Object, ByVal e As CommandEventArgs)

Dim thread As Integer

thread = Convert.ToInt32(Session("chatsess"))

con.Open()

cmd = New OleDbCommand("SELECT \* FROM [message\_threads] WHERE [thread\_id] = " & thread & "", con)

dr = cmd.ExecuteReader

If dr.Read Then

con.Close()

con.Open()

cmd = New OleDbCommand("DELETE \* FROM [messages] WHERE [thread\_id] = " & thread & "", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

con.Open()

cmd = New OleDbCommand("DELETE \* FROM [message\_threads] WHERE [thread\_id] = " & thread & "", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

End If

Response.Redirect("Messages.aspx")

End Sub

Publish Post:

Sub pub(ByVal Sender As Object, ByVal e As CommandEventArgs)

Dim commandArgsAccept As String = e.CommandArgument.ToString()

Dim value1 As String = commandArgsAccept.ToString

Session("possess") = value1

Dim poswri As Integer

poswri = Convert.ToInt32(Session("possess"))

con.Open()

cmd = New OleDbCommand("UPDATE [writes] SET [published] = True WHERE [writes\_id] = " & poswri & "", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Dim mysess As Integer

mysess = Convert.ToInt32(Session("idsess"))

Dim intDay As Date

intDay = Date.Now

con.Open()

cmd = New OleDbCommand("INSERT INTO activity(doer, type, date\_time\_written, post) VALUES('" & mysess & "', 'published a', '" & intDay & "', '" & poswri & "') ", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Response.Redirect("MyProfile.aspx")

End Sub

Happy Diary Search:

Protected Sub happy\_go(ByVal Sender As Object, ByVal e As EventArgs)

If Radiobuttonlist3.SelectedValue = "1" Then

Dim rmfrom As Date

rmfrom = DateTime.ParseExact(hapfromrm.Value, "dd'/'MM'/'yyyy", Nothing)

Dim rmto As Date

rmto = DateTime.ParseExact(haptorm.Value, "dd'/'MM'/'yyyy", Nothing)

Dim mywrites As String

mywrites = Session("idsess")

Dim may\_word As String

may\_word = happy\_words.Value

con.Open()

cmd = New OleDbCommand("SELECT \* FROM writes WHERE (([date\_written] BETWEEN #" & rmfrom.ToString("MM'/'dd'/'yyyy") & "# AND #" & rmto.ToString("MM'/'dd'/'yyyy") & "#) AND ([profile\_id] = " & mywrites & ") AND ([content] Like '%" & may\_word.ToString() & "%') AND ([happy] = True))", con)

overview.Style("Display") = "none"

diarysearch.DataSource = cmd.ExecuteReader()

diarysearch.DataBind()

con.Close()

ElseIf Radiobuttonlist3.SelectedValue = "2" Then

Dim rmfrom As Date

rmfrom = DateTime.ParseExact(hapfromrm.Value, "dd'/'MM'/'yyyy", Nothing)

Dim rmto As Date

rmto = DateTime.ParseExact(haptorm.Value, "dd'/'MM'/'yyyy", Nothing)

Dim mywrites As String

mywrites = Session("idsess")

Dim may\_not\_word As String

may\_not\_word = happy\_words.Value

con.Open()

cmd = New OleDbCommand("SELECT \* FROM writes WHERE (([date\_written] BETWEEN #" & rmfrom.ToString("MM'/'dd'/'yyyy") & "# AND #" & rmto.ToString("MM'/'dd'/'yyyy") & "#) AND ([profile\_id] = " & mywrites & ") AND ([content] Like '!%" & may\_not\_word.ToString() & "%') AND ([happy] = True))", con)

overview.Style("Display") = "none"

diarysearch.DataSource = cmd.ExecuteReader()

diarysearch.DataBind()

con.Close()

Else

Dim rmfrom As Date

rmfrom = DateTime.ParseExact(hapfromrm.Value, "dd'/'MM'/'yyyy", Nothing)

Dim rmto As Date

rmto = DateTime.ParseExact(haptorm.Value, "dd'/'MM'/'yyyy", Nothing)

Dim mywrites As String

mywrites = Session("idsess")

con.Open()

cmd = New OleDbCommand("SELECT \* FROM writes WHERE (([date\_written] BETWEEN #" & rmfrom.ToString("MM'/'dd'/'yyyy") & "# AND #" & rmto.ToString("MM'/'dd'/'yyyy") & "#) AND ([profile\_id] = " & mywrites & "))", con)

overview.Style("Display") = "none"

diarysearch.DataSource = cmd.ExecuteReader()

diarysearch.DataBind()

con.Close()

End If

End Sub

Sad Diary Search:

Protected Sub sad\_go(ByVal Sender As Object, ByVal e As EventArgs)

If Radiobuttonlist1.SelectedValue = "1" Then

Dim rmfrom As Date

rmfrom = DateTime.ParseExact(sadfromrm.Value, "dd'/'MM'/'yyyy", Nothing)

Dim rmto As Date

rmto = DateTime.ParseExact(sadtorm.Value, "dd'/'MM'/'yyyy", Nothing)

Dim mywrites As String

mywrites = Session("idsess")

Dim may\_word As String

may\_word = sad\_words.Value

con.Open()

cmd = New OleDbCommand("SELECT \* FROM writes WHERE (([date\_written] BETWEEN #" & rmfrom.ToString("MM'/'dd'/'yyyy") & "# AND #" & rmto.ToString("MM'/'dd'/'yyyy") & "#) AND ([profile\_id] = " & mywrites & ") AND ([content] Like '%" & may\_word.ToString() & "%') AND ([happy] = False))", con)

overview.Style("Display") = "none"

diarysearch.DataSource = cmd.ExecuteReader()

diarysearch.DataBind()

con.Close()

ElseIf Radiobuttonlist1.SelectedValue = "2" Then

Dim rmfrom As Date

rmfrom = DateTime.ParseExact(sadfromrm.Value, "dd'/'MM'/'yyyy", Nothing)

Dim rmto As Date

rmto = DateTime.ParseExact(sadtorm.Value, "dd'/'MM'/'yyyy", Nothing)

Dim mywrites As String

mywrites = Session("idsess")

Dim may\_not\_word As String

may\_not\_word = sad\_words.Value

con.Open()

cmd = New OleDbCommand("SELECT \* FROM writes WHERE (([date\_written] BETWEEN #" & rmfrom.ToString("MM'/'dd'/'yyyy") & "# AND #" & rmto.ToString("MM'/'dd'/'yyyy") & "#) AND ([profile\_id] = " & mywrites & ") AND ([content] Like '!%" & may\_not\_word.ToString() & "%') AND ([happy] = False))", con)

overview.Style("Display") = "none"

diarysearch.DataSource = cmd.ExecuteReader()

diarysearch.DataBind()

con.Close()

Else

Dim rmfrom As Date

rmfrom = DateTime.ParseExact(sadfromrm.Value, "dd'/'MM'/'yyyy", Nothing)

Dim rmto As Date

rmto = DateTime.ParseExact(sadtorm.Value, "dd'/'MM'/'yyyy", Nothing)

Dim mywrites As String

mywrites = Session("idsess")

con.Open()

cmd = New OleDbCommand("SELECT \* FROM writes WHERE (([date\_written] BETWEEN #" & rmfrom.ToString("MM'/'dd'/'yyyy") & "# AND #" & rmto.ToString("MM'/'dd'/'yyyy") & "#) AND ([profile\_id] = " & mywrites & "))", con)

overview.Style("Display") = "none"

diarysearch.DataSource = cmd.ExecuteReader()

diarysearch.DataBind()

con.Close()

End If

End Sub

Upload Photos WebHandler:

<%@ WebHandler Language="VB" Class="FileUploadHandler" %>

Imports System.Data

Imports System.IO

Imports System.Data.OleDb

Imports System.Web.SessionState

Imports System.Web.Security

Imports System.Web

Public Class FileUploadHandler

Implements IHttpHandler, System.Web.SessionState.IRequiresSessionState

Dim con As OleDbConnection = New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=C:\Users\Admin\Desktop\TimeGod Presentation\TimeGod-uncorrupt\TimeGod\_data.accdb")

Dim cmd As OleDbCommand

Dim cmd2 As OleDbCommand

Dim da As OleDbDataAdapter

Dim ds As New DataSet

Dim dr As OleDbDataReader

Public Sub ProcessRequest(ByVal context As HttpContext) Implements System.Web.IHttpHandler.ProcessRequest

Dim mysess As Integer

mysess = Convert.ToInt32(context.Session("idsess"))

Dim files As HttpFileCollection = context.Request.Files

Dim i As Integer

Dim file As HttpPostedFile = files(i)

If context.Request.Files.Count > 0 Then

For i = 0 To files.Count - 1

Dim fname As String = context.Server.MapPath("~/img/" + file.FileName)

file.SaveAs(fname)

con.Open()

cmd = New OleDbCommand("INSERT INTO picture\_library(user\_path, pic\_name) VALUES(" & mysess & ", '" & file.FileName & "')", con)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

Next

context.Response.ContentType = "text/plain"

context.Response.Write("File(s) Uploaded Successfully!")

End If

End Sub

Public ReadOnly Property IsReusable() As Boolean Implements System.Web.IHttpHandler.IsReusable

Get

Return False

End Get

End Property

End Class

**Testing**

**Testing Data Inputs**

**Validation Necessity:**

* It was necessary that the data entered by the user into their profile is accurate and correct.

(With this being a social networking site- you don’t want your profile to look foolish- do you?)

* Most parts of the validation coding is done inline itself- using jQuery and and a few plugins that are easy to find on the internet.  
  Once the validation patterns and rules were set, I bound the items to a submit button which only executes once the data-validations are stemmed true.

**Required Validations:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **ControlType** | **Operation** | **Expected Result** | **Actual Result** |
| First Name | TextBox | [a-zA-Z]+ | Allow | As expected |
| Null or otherwise | Display error | As expected |
| Last Name | TextBox | [a-zA-Z]+ | Allow | As expected |
| Null or otherwise | Display error | As expected |
| Date of Birth | TextBox | Not Null | Allow | As expected |
| Null or otherwise | Display error | As expected |
| E-mail Address | TextBox | ^[a-z0-9.\_%+-]+@[a-z0-9.-]+\.[a-z]{2,4}$ | Allow | As expected |
| Display error | As expected |
| Null or otherwise | Display error | As expected |
| Sex | TextBox | Not Null | Allow | As expected |
| Null or otherwise | Display error | As expected |
| City | TextBox | [a-zA-Z]+ | Allow | As expected |
| Null or otherwise | Display error | As expected |
| Country | TextBox | [a-zA-Z]+ | Allow | As expected |
| Null or otherwise | Display error | As expected |
| Username | TextBox | ^[\_A-z0-9]{1,}$ | Allow | As expected |
| Null or otherwise | Display error | As expected |
| Password | TextBox | Minimum of 6 characters | Allow | As expected |
| Null or otherwise | Display error | As expected |
| Confirm Password | TextBox | Data match password field (jQ) | Allow | As expected |
| Null or otherwise | Display error | As expected |
| Cell Phone | TextBox | (7|8|9)\d{9} | Allow | As expected |
| Null or otherwise | Display error | As expected |

**Testing Operations**

**Method:**

* Each action on every form was tested with self-made data as and when necessary.
* As including all the operations were not possible- this being a vast project and all, I’ve included all the operations which to me seem like an important part of the project.
* All the constraints are handled with error prompts if the user fails to provide the necessary inputs required.

**Tested Operations:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Page** | **Operation** | **Expected Result** | **Actual Result** |
| login\_sub | Registeration.aspx | Validated Click | Log the user in. | As Expected |
| Un-validated Click | Reload w/ error. | As Expected |
| like\_post | <multiple pages> | If Not Liked | Add to liked posts. | As Expected |
| If Already Liked | Prompt user. | As Expected |
| unlike\_post | Activity.aspx | Click | Remove post. | As Expected |
| comment\_post | <multiple pages> | Click | Add comment. | As Expected |
| red\_pro | <multiple pages> | Click | Redirect to profile. | As Expected |
| accept\_req | Activity.aspx | Click | Accept request. | As Expected |
| decline\_req | Activity.aspx | Click | Decline request. | As Expected |
| del\_acc | DelAcc.aspx | Validated Click | Delete account. | As Expected |
| Un-validated Click | Reload w/ error. | As Expected |
| stop\_f | Feelpals.aspx | Click | Stop user follow. | As Expected |
| sendmess | <multiple pages> | Click | Sends message. | As Expected |
| logout\_sess | <multiple pages> | Click | Logs out user. | As Expected |
| unf | <multiple pages> | Click | Stop following. | As Expected |
| sendmessnew | <multiple pages> | Click | Start conversation. | As Expected |
| rep\_pro | <multiple pages> | Click | Report profile. | As Expected |
| del\_pos | <multiple pages> | Click | Deletes post. | As Expected |
| pub\_pos | MyDiary.aspx | Click | Publishes post. | As Expected |
| happy\_go | MyDiary.aspx | Validated Click | Returns results. | As Expected |
| Un-validated Click | Prompts error. | As Expected |
| sad\_go | MyDiary.aspx | Validated Click | Returns results. | As Expected |
| Un-validated Click | Prompts error. | As Expected |
| search\_pals | Explore.aspx | Validated Click | Returns results. | As Expected |
| Un-validated Click | Prompts error. | As Expected |
| search\_posts | Explore.aspx | Validated Click | Returns results. | As Expected |
| Un-validated Click | Prompts error. | As Expected |

**Testing Correctness & Completeness**

**As previously seen in this section,**

* We went through a rigorous testing of many operations and field validations- leaving only a few aside since they only accomplish minor tasks that are handled efficiently regardless.  
  This defines & seals the testing correctness.
* Besides all of these test regimes, I’ve also taken care of adding to the website exception handling through the servlet code so that if something un-thought of is bound to go wrong, it’ll be handled by the exception.  
  This defines & seals the testing completeness.

**Not only that,**

* I’ve also taken care of the inline codes.
* Since a lot of data-sending had to be done via javascript, it was necessary to implement 100% correct and non-buggy tranisitions in order for smooth functioning.  
  You can see some of the examples in the code itself- or better- in the performance of the website.

**New Tools**

**And**

**Technologies**

**Learned/Used**

**New Tools Learned**

* **Advanced Javascript**The Repeater control is used to display a repeated list of items that are bound to the control. The Repeater control may be bound to a database table, an XML file, or another list of items.  
  Using the repeater was an absolute delight- it made working with bootstrap framework easy and provided the comfort of on-the-go inline coding which in return facilitated precise fetching results.
* **Servlets**

Asynchronous JavaScript and XML. In a nutshell, it is the use of the XMLHttpRequest object to communicate with server-side scripts. It can send as well as receive information in a variety of formats, including JSON, XML, HTML, and even text files.  
Learning to handle post requests and token requests from InstagramTM.

* **XML HttpRequest**

These controls remove the requirement to refresh the whole page with each postback, which improves the user experience.   
Using these panels along with the ScriptManagers helped over-buff some design flaws and made page loads and unnecessary post backs hassle-less.

**New Technologies Used**

* **Bootstrap Framework**Bootstrap makes front-end web development faster and easier. It's made for folks of all skill levels, devices of all shapes, and projects of all sizes.  
  Making prototypes with bootstrap was easy as breathing and allowed unparalled flexibility.  
  Although it didn’t fare well with Visual Studio’s Design Interface- it was worth hard-coding every single line of the website.
* **jQuery**Working with Bootstrap required me to go through a mandatory steep-learning curve associated with it’s functions that were governed by jQuery itself. Upon my increase of understanding, I absolutely fell in love with the simplicity of the language.  
  It helped lessen a lot of code-behind as making functions in conjuction to bootstrap elements was never this easier!

**System**

**Limitations  
 And**

**Dependencies**

**System Limitations/Restrictions**

* **Restrictions:**

Currently, the user is incapable of manipulating the Time-table by GUI. No drag and drop fucntions are coded as that would mean more of time being dedicated to develop a UI instead of the algorithm that had to be perfected.

**System Dependencies/Constraints**

* **Dependencies:**

The working and accuracy of time-table output depends on how well the user is able to interpret the wizard.  
(Although hints have been provided everywhere, a single mistake could result in an irregular time-table which should then be corrected by reverse-followign the steps.)

* **Constraints:**

A lot of validation has been put not only into the databases and forms but also upon the pages themselves.

This allows for excellent input of information as validations don’t allow for illegal inputs;  
Goes without saying that I’ve also implemented session states in order to regulate un-monitored non-activity of more than 30 minutes upon which it’ll statefully save your work and revoke it when you return.

**Future**

**Enhancements  
 And**

**Opportunities**

**Enhancements**

* **A drag-drop interface which auto-adjusts on each turn!** I’ve barely scratched the surface of what I originally intended this website to be made in.  
  Imporvements can be quantified as:
  + A perfectly capable GUI interface.
  + Customized .docx files.
  + Ability to manage multiple time-tables through the same account.
  + Better Wizard creation.
  + Optimized algorithm.

Realizing the limitless properties upon which these improvements can be made.

**Opportunities**

* **Research**If things go well and the development of this project doesn’t make into my yet-to-do list, I might as well form a team and go fully with this website.  
  Even Facebook doesn’t facilitate it’s users with absolute freedom towards exploration- something that TimeGod excels at. I’ve taken a carefully accomplished thought process of merging the best parts of all the social networking services and blended it into a easy-to-user interface with a target audience of the most artistic following.
* **College/School Server Hosting**All else aside, I would love to upload the site on the servers of my old schools and colleges just to learn more about time-table requisites and user statistics. Alongside, providing a free service for betterment of work-flows.

**Bibliography  
 And**

**References**

**Book References:**

* The Complete Reference Visual Studio.NET
* Visual Basic .NET Black Book
* Software Engineer’s: A Practitioner’s Approach by Roger Pressman
* System Analysis, Desing and Introduction to Software Engineering by Partha Sarthi

**Internet References:**

* The Almighty Google (www.google.com)
* Stack Overflow for Inquiries (www.stackoverflow.com)
* GitHub for resources ([www.github.com](http://www.github.com))
* SourceForge for resources (www.sourceforge.net)
* ASP Snippets ([www.aspsnippets.net](http://www.aspsnippets.net))
* YouTube ([www.youtube.com](http://www.youtube.com))
* CodeConverter for understanding non-primary languages (<http://converter.telerik.com>)