Software Development Group

COMP 730/830, Spring 2017

Professor Jonas

LEADERSHIP ROLES

Software Development Managing Architect: [redacted]

Software Development Quality Assurance Specialist: [redacted]

Software Development Communications Specialist: Patrick McElhiney

DEVELOPMENT ROLES

Graduate Software Development Engineer, Level 1: [redacted]

Graduate Software Development Engineer, Level 2: Patrick McElhiney

Graduate Software Development Engineer, Level 3: [redacted]

Software Development Engineer: [redacted]

Lead OOP User Interface Communications Specialist: [redacted]

Lead OOP Back End Systems Specialist: [redacted]

CONTACT INFO

[redacted]

Patrick R. McElhiney **(603) 742-5112** **prr22@wildcats.unh.edu**

[redacted]

Current Task List

Task Description:	**Assigned To:**	**Due Date:**
Revise UML Diagrams	[redacted]	03/22/2017
• Update Class Diagrams		03/22/2017
• Create Object Diagrams		03/22/2017
Clean up the Documentation (on-going)	[redacted]	05/02/2017
Create Fully Working Demo For 3/22/2017 Class	[redacted]	03/22/2017
Reports on Various Types of Searching Algorithms		03/22/2017
• Fuzzy Search	[redacted]	03/22/2017
• Nearest Neighbor Search		03/22/2017
• Hamming Distance Function		03/22/2017
Get Demo Working on UNH Server	Everyone	03/29/2017
• This will allow other groups to work on it.		
Get Redmine Working	[redacted]	03/29/2017
• Either on Patrick's Linux box or on [redacted]'s Azure account.		
Revise Coding with Algorithms	Everyone	04/05/2017
Come Up with QA Test Questions	[redacted]	04/05/2017
• Unit Testing		
Integrate Problems Found into Code	Everyone	04/12/2017
Perform Testing Again	[redacted]	04/19/2017

Notes Regarding Weekly Assignments

If your name is not listed to finish a specific task during any week, your job is to assist who is leading the project operations during that week per the schedule. Always stay in contact with the Architect, and the person who is responsible for delivering the finished item(s) by the specified due dates.

Note that if you do not perform each week, you will be graded down per the group's grading of you – so be sure you are on top of everything.

If any part of our project responsibilities is failing – it is ultimately the responsibility of the other Engineers to pick up the pieces and take over.

- **Meet Deadlines** – make sure all work is completed by due date(s)
- **Delegate** – be sure you split up the work load among all group members.
- **Be Prepared** – make sure all group members have the necessary software and assigned tasks to stay busy each week.

If you need help, please ask a fellow Engineer, or a Graduate Student.

If for some reason, you are unable to complete a task, make sure you send all the materials to both the Architect ([redacted]), Quality Assurance ([redacted]) and Communications (Patrick M.).

Please note that if you are a Graduate Student, you have additional responsibilities as defined in the Job Descriptions, and per any other agreed to schedule or timeline per the group.

- Architect should coordinate and manage all aspects of the team / group.
    - All members of the group should stay in contact with the Architect, every week, multiple times per week, to provide updates, hold meetings, etc. as required per the project.
- Quality Assurance should develop questions and test parameters on a weekly basis, and answer the questions to determine if the quality of the workmanship is up to par.
- Communications should keep all members on the same page, not just with what our group is doing, but what all the groups are doing.
    - Undergraduate and Graduate Students are encouraged to reach out to Patrick M. on a weekly basis, multiple times if possible, to get updates, request information, request help, etc.
    - They should also not necessarily wait for Patrick M. to respond – it is not off limits to contact any member of the class regarding any work that is being done. If you need to contact someone in Front-end (GUI) or Database, you should be able to do that on your own, too – but just keep Patrick M. in the loop.
- Each group member should take responsibility for the entire project.

EXPECTATIONS OF TEAM MEMBERS

Per the Class Syllabus,

All Team Members are Expected To:

- Assume Developer Roles.
- Write the User Requirements of a Real-World Team Project, **Intensively**, through:
  - **C**apture,
  - **A**nalyze,
  - **R**efine,
  - **D**ocument.
- Participate in Weekly Development Activities, by Working in Teams to Build Models of a Real-World System, Including:
  1. Requirements Elicitation and Analysis,
  2. System and Object Design,
  3. Implementation and Testing,
  4. Project and Configuration Management,
  5. Infrastructure Maintenance,
  6. System Deployment to the End User.
- Deliver a Proof-of-Concept, or Prototype of Me&You.
- Carry on the Architectural Task of System Analysis.
- Plan for, Manage, and Mitigate Risk Factors the Team Might Encounter.
- Improve Personal and Interpersonal Communication through Interaction with Team Members and a Real Client.

JOB DESCRIPTIONS

SOFTWARE DEVELOPMENT MANAGING ARCHITECT

- Responsible for all software development designing / software development engineering tasks.
    - Develops the overall software development design / software development structure, and disseminates information about it to task other group members to develop the IP assets and configurations that will be needed for the Software.
    - Develops Use Cases and Use Case Diagrams to lead the Software Development Group in the development of the software for Me&You.
- Manages the Software Development Group on a weekly basis, assigning tasks to Graduate and Undergraduate students.
    - Manages / Updates the Job Descriptions for the entire Software Development Group, assigning additional work / sub-blocks as needed based on the overall needs of the Software Development Group.
    - Facilitates the development of the core infrastructure with the Database Group and Front-end (GUI) Group, based on the architecture of the Software Development design.
- In charge of packaging up everything for Software Development Group for the next Semester.
- Maintain communication with the client (*Professor Jonas*).
- Keeps Graduate Logs Starting on March 1st, 2017.
- Coordinates the Quality Assurance process with the Software Development Quality Assurance Specialist.
- Communicates with other Architects regarding the Project Scope and Configurations of the overall Project Software Development Core.
- Coordinates communications with the Software Development Communications Specialist, and with other groups per and through the Software Development Communications Specialist.
    - Communicates the Software Development Design and Engineering Scheme to Front-end (GUI) and Database Groups
- Acts as Graduate Software Development Engineer Level 3, in support of the Software Development Architect's goals and tasks on a weekly basis.
    - Sets up Server for Software, and imports Software to Server.

JOB DESCRIPTIONS (continued…)

SOFTWARE DEVELOPMENT QUALITY ASSURANCE SPECIALIST

- Responsible for all software development quality assurance processes, including the development and fulfillment / verification of all Quality Assurance questions dealing with the Software Development Group.
    - In charge of all Requirements Elicitation processes
        - Refines lists of attributes based on input from the Front-end (GUI) and Database Groups, as well as the Software Development Group.
    - Coordinates with other QA Specialists, to streamline the process of developing QA procedures and limitations for each of the groups' Quality Assurance and separately for Software Development group with input from the other QA Specialists.
    - Maintains systems associated with Source Code Control, and ensures proper documentation in Source Code, in addition to QA procedures dealing with the Source Code
        - Double checks the Source Code Control processes and procedures
- In charge of the maintenance of all documentation dealing with the Software Development Group, working with the Communications Specialist to disseminate the information to other groups on a timely basis.
- Keeps Graduate Logs Starting on March 1st, 2017.
- Acts as Graduate Software Development Engineer Level 1, in support of the Software Development Architect's goals and tasks on a weekly basis.

JOB DESCRIPTIONS (continued…)

SOFTWARE DEVELOPMENT COMMUNICATIONS SPECIALIST

- Coordinate all communications between Graduate Students with weekly Zoom meetings (*Tuesdays 4:00PM – 6:00PM*), before class meetings (*Wednesdays 3:30PM – 5:30PM*), and after class meetings (*Wednesdays 8:30PM – 9:30PM*)
    - Between Front-end (GUI) Group and Software Development Group
    - Between Software Development Group and Software Development Group
    - Between Front-end (GUI) Group and Software Development Group
- Maintain communication with the client (*Professor Jonas*) and communicate and coordinate all contributions to the project, overall.
    - *"Improve personal and interpersonal communication through interaction with team members and a real client."*
- Ensures that all groups are on the same page, per the communication requirements between different groups – including between Architects, between QA Specialists, and between Engineers.
- Coordinate all communications between Graduate Students and Undergraduate Students in the Software Development Group (*Starting in March 2017 – May 2017*)
- Develops and Maintains Timeline Plan, based on the Deliverables Identified by the Software Development Architect, and the Software Development Quality Assurance Processes as Identified by the Software Development Quality Assurance Specialist
    - *"Plan for, manage, and mitigate risk factors the team might encounter."* in terms of the planning of deliverables, and the timing of the project, by managing the group on a ramping-up basis from week to week using communications and Microsoft Project management tool with Gantt Chart
    - Keep track of how many hours each student has spent towards the project in Microsoft Project, or otherwise in some type of table or chart such as Excel
- Updates the Wiki page on a weekly basis.
- Keeps Graduate Logs Starting on March 1st, 2017.
    - Develops outline form for Graduate Student Logs, and disseminates this to every Graduate Student by March 1st, 2017.
- Acts as Graduate Software Development Engineer Level 2, in support of the Software Development Architect's goals and tasks on a weekly basis.
    - Develops an Outline Form for the Software Development Group Documentation, and disseminates this document to the Software Development Group, as well as the other groups as an example for them to copy and use with their own information.

JOB DESCRIPTIONS (continued…)

LEAD OOP USER INTERFACE COMMUNICATIONS SPECIALIST

JAVA, C#, PHP, ETC.

- Designs the code that interfaces the back-end procedure calls system (or data sorting back-end programs) with the front-end's code, which is written primarily in PHP, with HTML / CSS experience needed.
- Works directly with the **Front-end (GUI) Architect** ([redacted]) and the **Front-End (GUI) OOP Front-End Developer** ([redacted]) to facilitate the integration of the back-end database with the Front-end User Interface through HTML/CSS/PHP/MySQL Interface
  [redacted]

LEAD OOP BACK END SYSTEMS SPECIALIST

JAVA, C#, MYSQL, AUTOMATED TASKS & PROCESSES IN WINDOWS SERVER 2008, ETC.

- Designs the code that interfaces the database procedure calls with data sorting and manipulation, with an emphasis on:
  - Data Efficiency,
  - Security, and
  - Automation of Tasks.

JOB DESCRIPTIONS (continued…)

SOFTWARE DEVELOPMENT ENGINEER

JAVA, C#, MYSQL, ALGORITHMS FOR SEARCH/MATCH OF DIFFERENT STRINGS

- Researches and implements various search algorithms that allow for matching between multiple strings that are similar with respects to some properties and dissimilar with respects to other properties
    - Linked Lists, or relational characteristics between multiple strings based on their relation to other strings, such as:
        - Matching First Names based on similarities, such as those linked in a relational database, i.e. (http://www.behindthename.com/)
            - **Hierarchical Relationship** – such as by entering all of the different combinations of similarities for each name.
                - **Variants**
                - **Diminutives**
                - **Other Languages / Cultures**
            - **User Submissions** – such as having the option for the user to submit other spellings of the person's name in the attributes of the search query
        - Matching Strings based on their misspelled version
            - **Relational Database**
            - **Fuzzy Search**
            - **"Naive String-matching Algorithm"**
            - **Rabin-Karp Algorithm**
            - **String matching with Finite Automata**
            - **The Knuth-Morris-Pratt Algorithm**
- Writes code that makes the Engine work for the searching on the website, using the algorithms to compare different values for different attributes, and match them based on a weight that is required to define a match

TEAM RESOURCES

- Experience
  - [redacted] (Architect) – works as a systems designer at Comcast. He has experience with C# and .NET, and pays for a Microsoft Azure account, in which he has used to create a working demo of our platform.
  - [redacted] (Quality Assurance) – has a B.A. in Computer Science, experience with Unit Testing and Integration Testing
    - Language Experience: OOP using Java, dynamic web development language using HTML, CSS, JavaScript, MySQL. It is easy for him to learn and master either C# or .NET if needed.
    - He should be able to help in identifying efficient algorithms to do matching process.
    - Also, make our part communicate with user interface (UI) and back-end.
  - **Patrick McElhiney** (Communications Director) – runs his own Marketing Firm, and has experience programming in Java, Python, Shell Scripting, and several other languages – mostly based on Linux deployments. He has a Linux shell that is available if the Software Group needs a Linux box to test out anything.
  - [redacted]
    - Experience with C#. Algorithms.
    - What other background / experience?
  - [redacted]
    - Experience with C#.
    - What other languages does he have experience with?
    - What is his background in programming and software engineering?
  - [redacted]
    - Experience with C#.
    - What other languages does he have experience with?
    - What is his background in programming and software engineering?
- Schedules (*When are people available?*)
  - [redacted]
    - **Tuesday** – 4PM to 5PM.
    - **Thursday, Friday, Sunday, and Monday** – 7PM to 10 PM
    - **Sunday** – 8AM to 12 PM (Tentative)
  - [redacted]
    - **Monday to Thursday** – 8AM to 4PM.
    - **Friday** – 8AM to 1PM.
  - Patrick R. McElhiney
    - **All Days, All Times** – Until Further Notice
  - [redacted]
    - Need availability schedule.
  - [redacted]
    - Need availability schedule.
  - [redacted]
    - Need availability schedule.

TEAM RESOURCES

- Computer Programs
  - [redacted] has Microsoft Visual Studio.
  - [redacted]
    - What programs does he have / need?
  - Patrick McElhiney has:
    - **Microsoft Project** (for Gannt Charting)
    - **Microsoft Visio Professional** (for UML / Crow's Feet Diagrams)
    - If necessary, could re-subscribe to Microsoft Action Pack and get Microsoft Visual Studio / MSDN licenses for up to 3 programmers. It costs a Microsoft Registered Partner $475.00/year for this package. It also comes with other Microsoft Software licenses for almost all Microsoft Software.
  - [redacted]
    - What programs does he have / need?
  - [redacted]
    - What programs does he have / need?
  - [redacted]
    - What programs does he have / need?
  - Notepad++ will be needed.
  - Eclipse Neon is a good tool for programming for Java, but also for C, C++, COBOL, D, Fortran, Haskell, JavaScript, Julia, Lasso, Lua, NATURAL, Perl, PHP, Prolog, Python, R, Ruby, Rust, Scala, Cloujure, Groovy, Scheme, and Erlang.
- Team Collaboration Resources
  - Slack – OOSE Slack, setup by [redacted]
    - Download Slack for Desktop
    - oose-team.slack.com
  - Zoom
    - zoom.unh.edu
    - Patrick McElhiney's Meeting ID: 308-196-7750
    - Join from PC, Mac, Linux, iOS or Android: https://unh.zoom.us/j/3081967750
- Books

  Patrick M. has loaned several books out to students in COMP 830/730. Please ensure that they are returned when they are no longer needed for the project. If additional books are needed, you can check them out at the Library, or ask Patrick M.

Loaned On:	Loaned To:	Book Loaned:
2/22/2017	[redacted]	Redmine Book
3/1/2017	[redacted]	Java / MySQL Book
3/1/2017		PHP Security
3/1/2017	[redacted]	Apache Security
3/1/2017	[redacted]	Linux Box Security with IDS / Snort

MEETINGS SCHEDULE

All Groups, All Members

→ Unless Specifically Excused Due to Schedule Conflict – Must Notify Patrick McElhiney in Advance

Weekly Zoom Meetings:

Tuesdays 4:00PM – 6:00PM

Graduate Students – Starting 2/21/2017 through 5/14/2017

⇨ All Graduate Students Also Meet on 3/14/2017 During Spring Break

Undergraduate Students – Starting 3/21/2017 through 5/14/2017

- zoom.unh.edu
- Patrick McElhiney's Meeting ID: 308-196-7750
- Join from PC, Mac, Linux, iOS or Android: https://unh.zoom.us/j/3081967750

Software Development

Before Class **Meetings:**

Wednesdays In-Person 3:30PM – 5:30PM

Graduate Students – Starting 2/22/2017 through 5/10/2017

⇨ We Will Not Meet in Person on 3/15/2017 During Spring Break

Undergraduate Students – Starting 3/22/2017 through 5/10/2017

Front-end (GUI) & Other Groups as Needed

After Class **Meetings:**

Wednesdays In-Person 8:30PM – 9:00PM

Graduate Students – Starting 2/22/2017 through 5/10/2017

⇨ We Will Not Meet in Person on 3/15/2017 During Spring Break

Undergraduate Students – Starting 3/22/2017 through 5/10/2017

Final Stretch

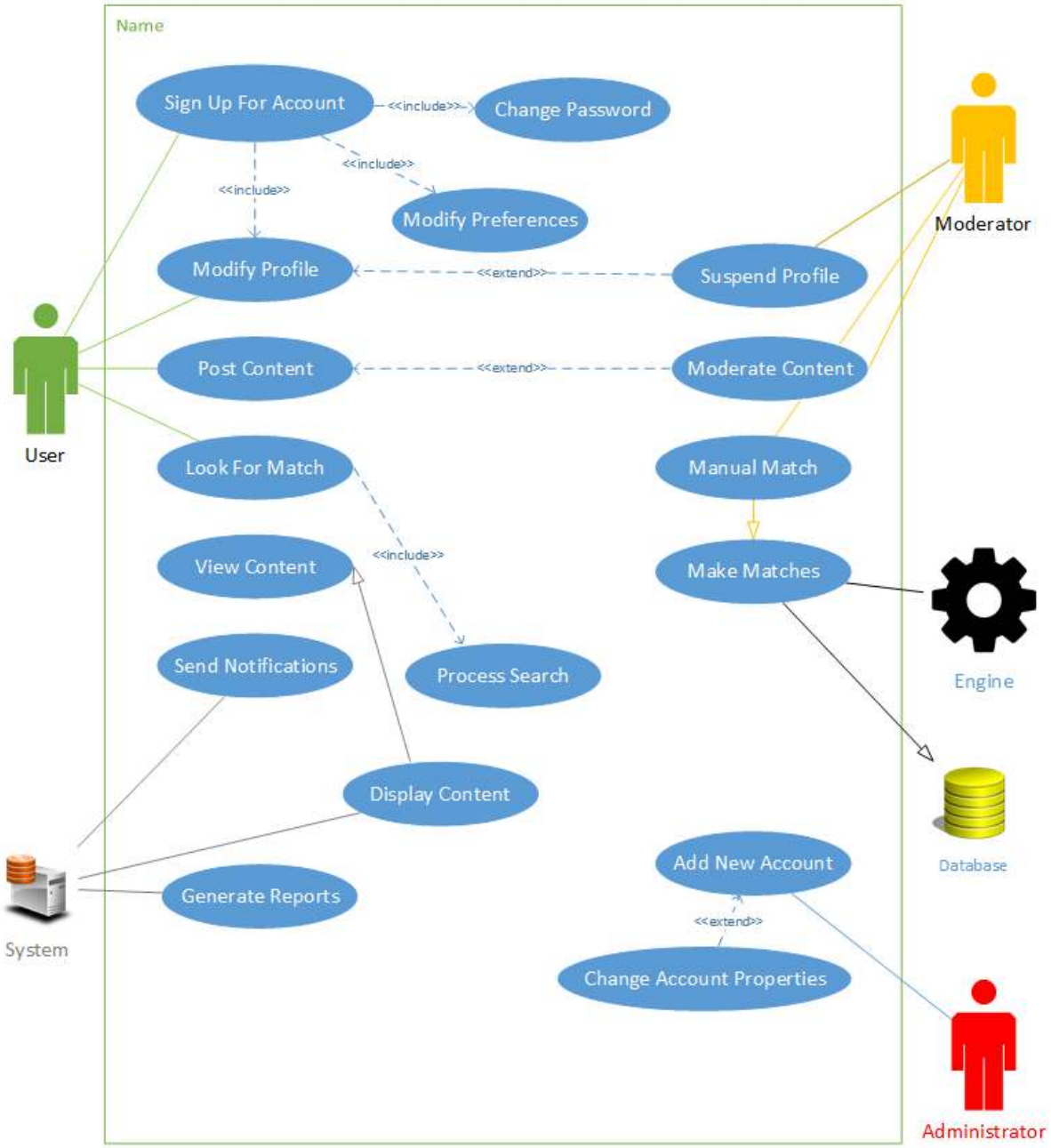
Last Week TBD **Meetings:**

May 11th through May 15th (Please Request Vacation Time in Advance if Possible)

- ⇨ All Groups, All Members – Meetings TBD
- ⇨ Separate Groups, All Members – Meetings TBD
- ⇨ Separate Groups, Graduate Students Only – Meetings TBD
- ⇨ All Graduate Students, All Groups – Meetings TBD
- ⇨ Communications Director, Select Students *(for Makeup Work)* – Meetings TBD
- ⇨ Architects, Professor Jonas – Meetings TBD

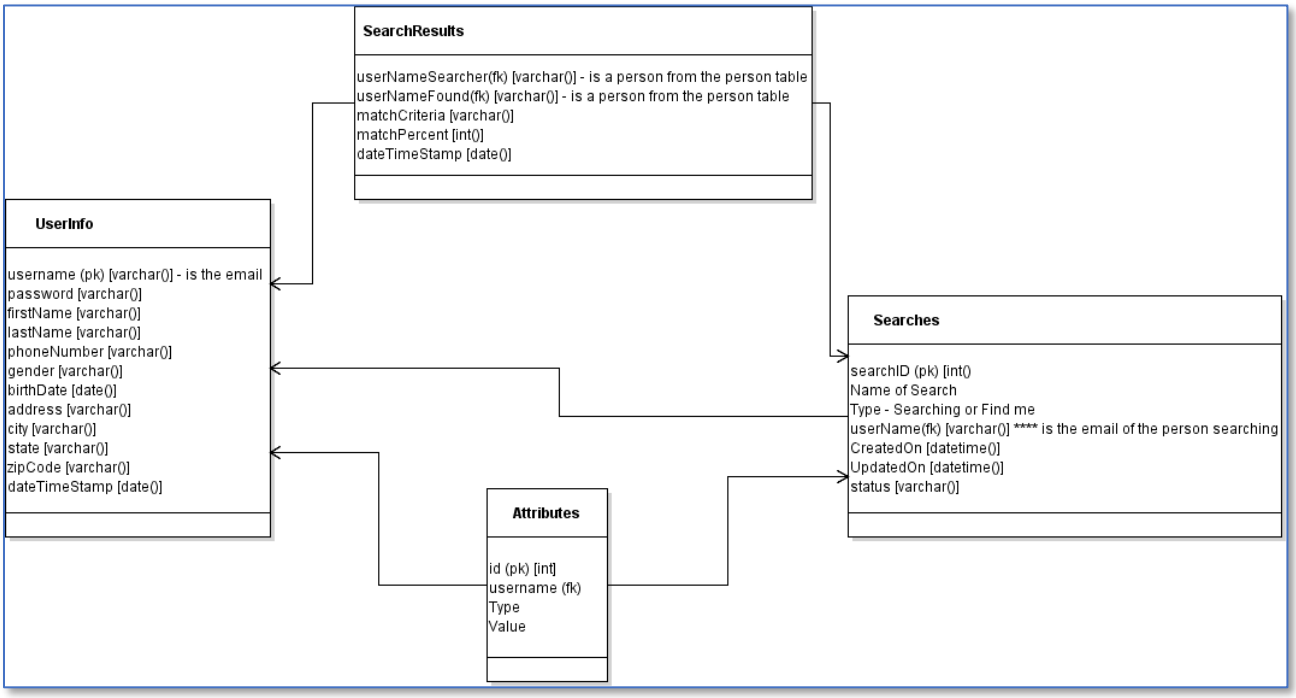
Software Development UML Diagrams (Week 2)

Me&You – General Use Case Diagram



Tables Needed for Searching

Based on "***MeAndYou Project Master Attributes***" from Spring 2016
(https://foss.unh.edu/projects/index.php/comp730:MeAndYou_Project_Master_Attributes) with some modifications



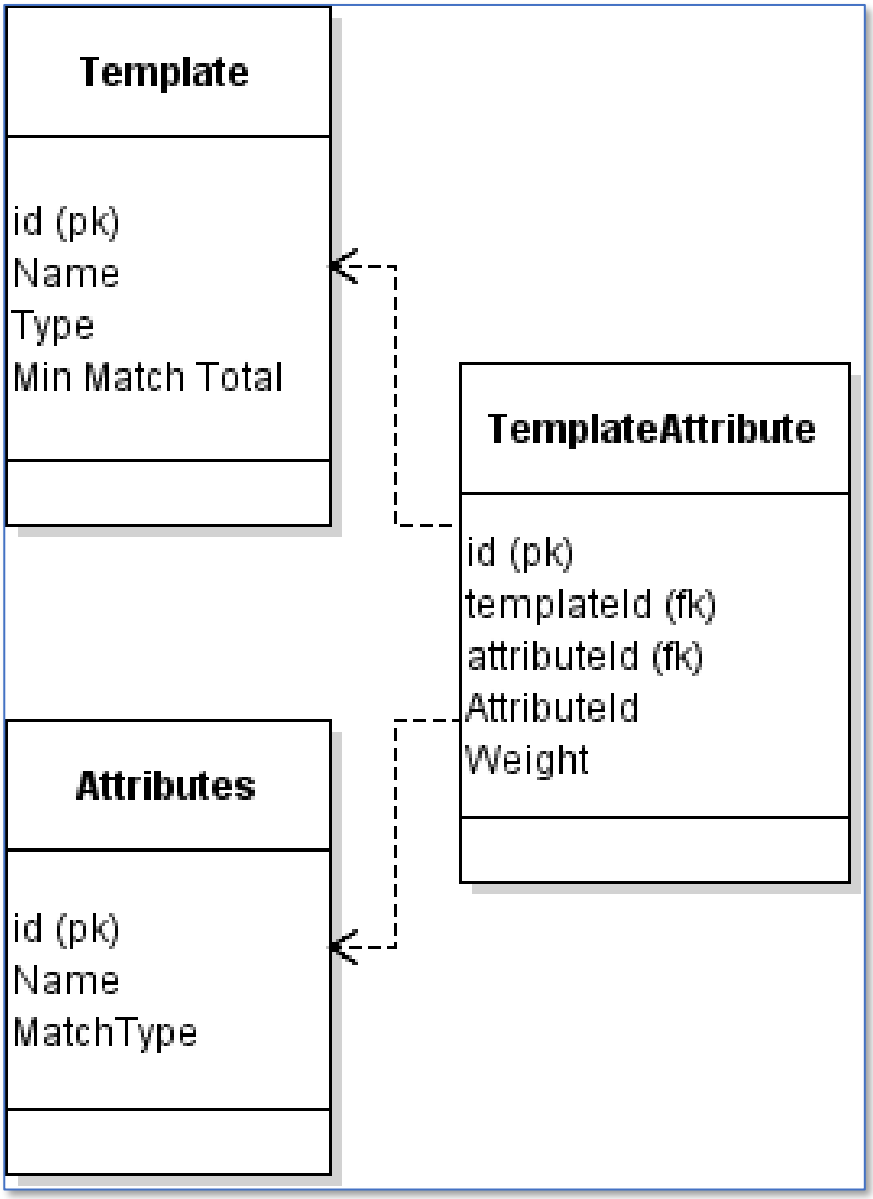
- The user enters their personal information
- The user can then add their attributes for the "Crush" search. These attributes are for someone finding them. The attributes that are required are determined by the Template. This is used to display the form for data entry.

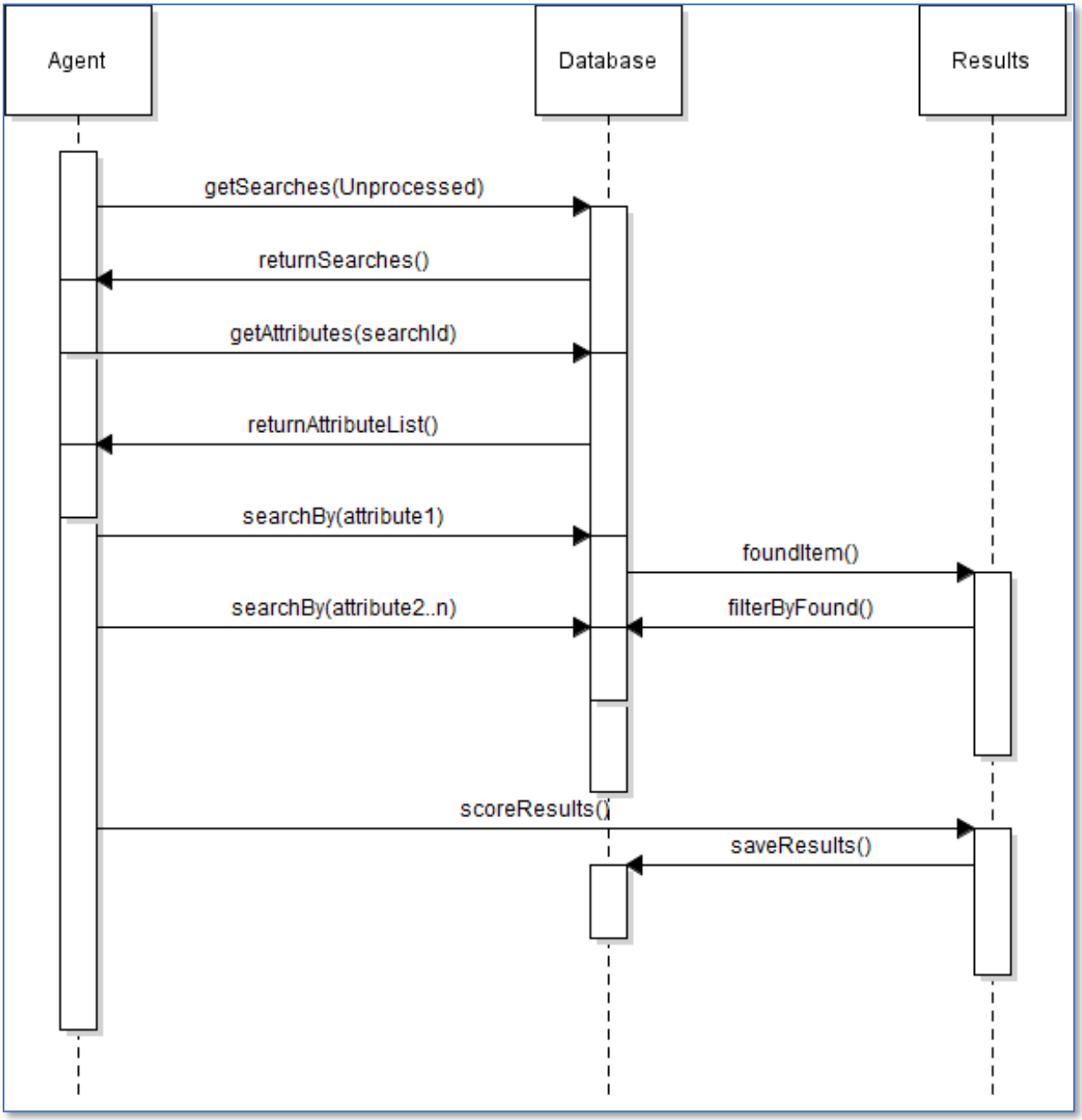
Template

The template allows the application to be expandable to other types of searches and does not require altering the database if additional attributes need to added.

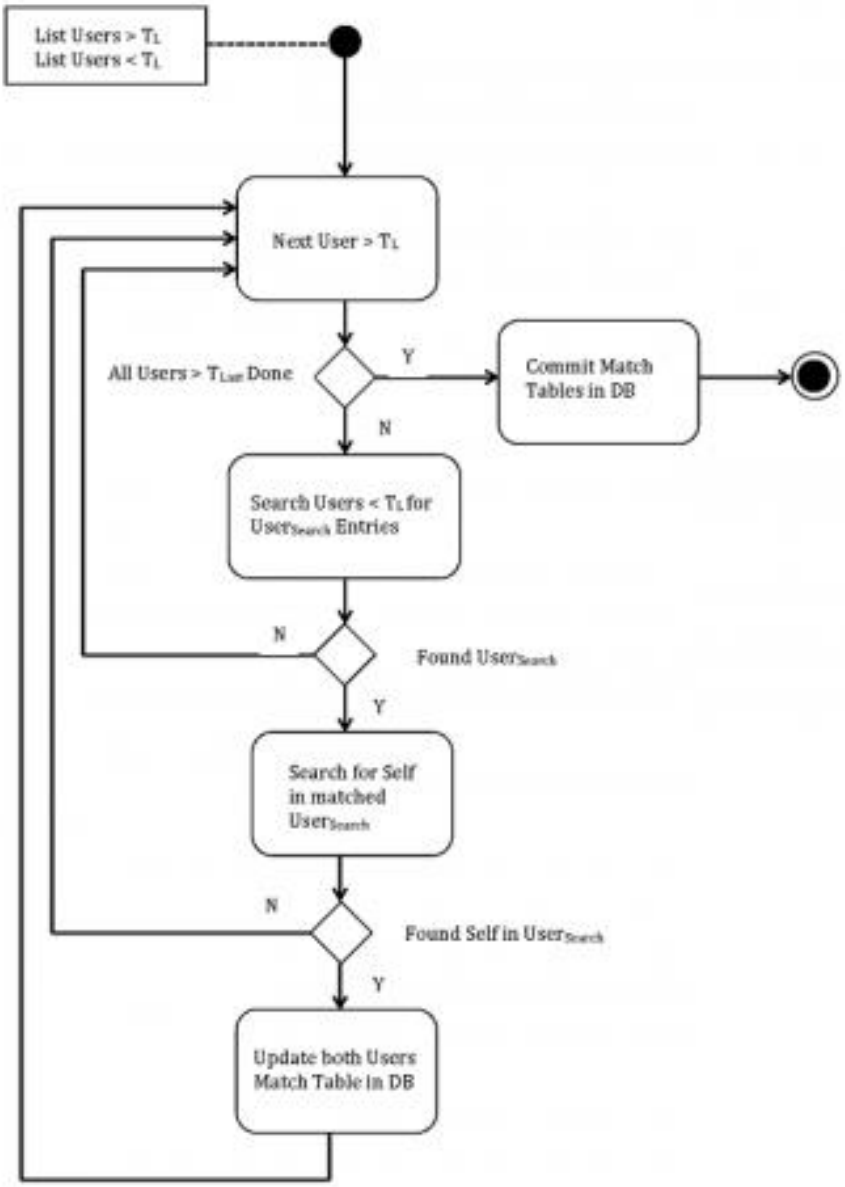
The original DB diagram had a Phone and Email table. This means we would also need a school and associations if you wanted to search on those attributes.

This model allows for flexibility.




Agent
Database
Results
getSearches(Unprocessed)
returnSearches()
getAttributes(searchId)
returnAttributeList()
searchBy(attribute1)
foundItem()
searchBy(attribute2..n)
filterByFound()
scoreResults()
saveResults()

Search Diagram without Individual Entries Breakdown:



Concepts for Processing Information (Week 2)

Fuzzy Search (http://www.tsjensen.com/blog/post/2011/05/27/Four+Functions+For+Finding+Fuzzy+String+Matches+In+C+Extensions)

Nearest Neighbor Search - Ask Steve Saunders

Algorithms: for Hamming distance function (https://en.wikipedia.org/wiki/Hamming_distance)

Search Agent

Assumes:

Search for the Attribute that has the highest Match weight. For instance, if Phone Number must match (tolerance of 1), then search for that first.

Processing

- Agent queries Searches table for unprocessed with type = search
- Gets a list to process
- For each item in list,
    - Get the attributes to search for sorted by tolerance.
    - For each attribute in list
        - Find the closest match (1 = perfect….0.nnnn for close enough). (The template will determine what is acceptable for close enough).
        - If item meets criteria, Save the match value to results array
            - What if multiple matches?

For next item to search for, only compare to items found in the list.

This will narrow the number of records to search across to a smaller subset

For the results returned, sum the attributes, and return the list that meet min criteria

Software Development Notes (Week 2)

Matches:

If match, capture the UserNameId of the Searcher and the UserNameId of the Found and the Weighted Total. Store to the SearchResults table.

Searches:

Header record to hold:

Search Name

UserNameId that submitted

Type: Search For or Find me

CreatedOn

UpdatedOn

Details to hold the attributes based on the template

Template determines the rules for Crush:

Attributes to match

% of match to determine a match

Weighting of the value

Timing:

Multiple threads

Running continuously

Need to Discuss (Week 2)

Not too sure what this means:

Set a percent threshold to determine if match table for users should be updated. Order match tables to handle greater than 5 matches for users. If percent is too low, then a lower match can be used and order table with highest first.

> Load data in 'chunks' to data structures in memory. Chunks could apply to both newer tables and the tables searched previously.
>
> Consider possibly use name lookup tables to handle alternate names used. An example is Jon Smith's first name may be Jon, John or Jonathan [update: this may be better for first letters only].

Prior Class Logic (Week 2)

Get list of updated tables

While updated tables not done:

    Load batch of updated tables

        While all old tables not searched:

            Load batch of old tables

            Perform matching algorithm (this will expand)

                For one updated user table match to batch of old tables

                    For each field, prepare and perform </u>near match</u>

                    Compile results to determine if match

                Update match tables

            Commit to database before next batch (note – add to old table load?)

Questions from Database Group (Week 3)

Software Development

- Need number of characters allowed for each field, and information also needs to be shared with Front-end.

REQUIREMENTS ELICITATION AND ANALYSIS

Project Requirements

- Develop prototype website to handle 20k-25k users within 8 weeks of 3/22/2017
    - It must work
    - Last year's class did a demo. I.e. it's possible
        - It must be more than a Demo
        - Original Title "Thinking of you…"

Application Domain

- A user must be able to connect with someone that isn't necessarily a stranger, but they don't necessarily have all their personal information.
- Categories are:
    - **Crush** – 2 week waiting period
        - Enter their e-mail address.
        - Create a level of trust that isn't already there.
    - **Love** – 1 year waiting period *(Long Lost Love)*
        - You must know the person.
        - "Navigating Landmines"
        - Put as much information about you and them.
        - It's based on information, not login
        - For Long Lost Love, they can't use the Crush function?
    - **Family** – 6 month waiting period. *(Distant Family Member, Long Lost Family Member?)*
    - **Friends** – 3 month waiting period. *(College Friendship, Long Lost Friends?)*
- Who defines what the required, and optional attributes, should be for each match type?
    - Personality?
        - What personalities are similar, in terms of their weights in relation to each other?
    - Hobbies?
    - Activities?
        - What will they do on their date / meeting?
- What is the maximum number of searches that can be performed by a user?
    - Should it be limited to a specific max. frequency?

Solution Domain

- Auto-selection of sex for match when Crush or Love, if sexual orientation is set in user profile attributes.
- Ability to set both required and optional field combinations for each type of match, and "one or the other" requirements. I.e.
    - **Crushes**
        - **Required:** First Name, Gender
        - **One of the Other:** Phone Number, E-mail
        - **Optional (3 Required):** Hair Color, Last Name, Birthday, Zodiac Sign, Place of Meeting, Hobbies, Personality Type(s)

REQUIREMENTS ELICITATION AND ANALYSIS

Non-Functional Requirements

- The website must be unique in the way it operates, and it must be usable.
- Reliability
    - Make sure you don't make any last-minute changes before a live Demo of the Software / Backend.
    - Avoid creating problems with your code that crashes the server. I.e. infinite loops
        - Always have a timeout set on the script / applet you're working on.
        - If the server isn't responding:
            - Front-end – just close the window. PHP stops executing when the session ends.
            - Backend (Engine) – kill the process that's associated with the program.
    - Consider that other people may be trying to work on the server at the same time.

Documentation Requirements

- Clean up past class's documentation and keep better notes on everything that we do, contemplate, and decide.
    - Conformity to the documentation, i.e. develop templates and distribute them to team members
- Graduate Students create logs that are used for grading purposes.
- Everyone must share the project.
    - There should be no "compartmentalization".
- Don't put Intellectual Property in the Cloud, i.e. Google Drive
    - "Billion Dollar Product" – like "Cold Fusion", companies have stakeholders, will take the Intellectual Property if it is out there
    - No sharing of Intellectual Property on Slack

User Profile Requirements

- User can't have multiple profiles
- Require User Login to use Me&You Website
- Modify User Profile View will be user friendly - you can enter all the Profile Attributes on the same page.

Security Brainstorming

- Don't allow empty or incomplete searches to enter the Database

SYSTEM DESIGN

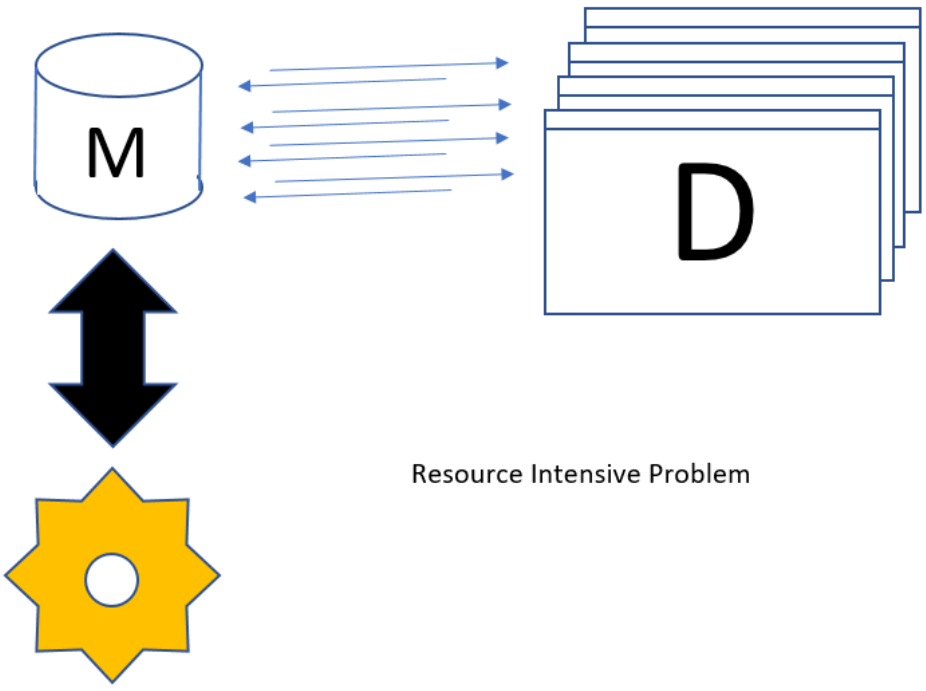
System Design

- Hardware Mapping
    - Power Blade 510
        - 2x Xeon Processors, each at 2.93Ghz
        - 64GB of RAM
        - Runs Windows Server 2008 R2 Natively
            - On 3/9/2017, Professor Jonas said that Windows Server was not activated.
        - Operating System: To Be Determined…
            - Linux VPS?
            - Windows Server VPS? (*Last year's class used Windows Server*)
        - Currently runs 6 VMs, 2 are for Me&You
        - There are currently network problems with the server that are trying to be worked out as of 3/9/2017. A network card was pulled out. The issue seemed to fix its self for a little while, and then it broke again. Professor Jonas said this issue may be related to the software activation not having occurred, or the system being corrupt and thus lost its Windows Activation.
            - One way to currently get into the system is to log in to the Host system, and then connect locally to the VM for Me&You.
        - The server originally cost $20,000 when first purchased.
- Domain Names: meandyou.*, main domain **meandyou.us**
- Ideally to deploy at scale (Millions of users), we would implement utilizing a Platform as a Service (PAAS) for the Website, which comprises the front-end of the system including the Graphical User Interface (GUI). This will allow for meeting growing demand and increased market share.
- Database can handle growth?
    - No problem for up to 25k users, up to 10k searches - or 5,000 potential matches being processed.
    - Multiple Agents Searching Simultaneously
        - No problem, if the cells are not being written to while they are being read. There are settings in this regard, such as making the process wait until the locking is released on the specific cell that is being written to. It can allow the old information to be read during the write if they happen at the same time. Since we aren't designing a 911 system or something that needs the latest information always, the locking while writing should suffice.

OBJECT DESIGN

Object Design

- Stack Design (CIS Majors)
- Apply the Software Engineering Principles (M.S. IT Students – Systems Integrations)
  - M.S. IT students are not expected to be experts in programming, but they should be.
- Make refinements to existing project, not necessarily significant changes.
  - If you have a better idea, we can use it, but you'll have to make a good argument.
- The engine that gets data from the database, updates the GUI
  - Make engine efficient, to resolve the resource intensive problem
  - Shrink the algorithm to be more efficient.
- Text Matching – convert to UPPER() or LOWER() before testing strings against each other.
- Numeric Matching – convert to integer in source code before testing.
- **Scoring** – weighting of the matches?
  - There should be individual weights on the individual attributes that are compared for a match percentage, such as by Fuzzy Search or another method.
  - There should also be an overall weight, such as the average, or weighted average of the weights of each individual attribute that has been scored.



IMPLEMENTATION AND TESTING

- Unit Testing
  - Code will have unit tests for all bugs, and modules
  - Not 100% code coverage
- User Testing
  - QA Team will develop test scenarios
  - Add UI Tests
    - Automated tool to perform standard user tests to allow for quick testing and validation
- Acceptance Testing
  - Key users will perform acceptance Tests to
- We have a responsibility to do a better job than last year's class did.

PROJECT AND CONFIGURATION MANAGEMENT

Project Management Process

- Redmine
  - Manage Users of System and Roles
    - Managers
    - Lead Developer
    - Developers
    - QA
    - Business Analysts
  - Bug Tracking
    - Ability to capture issues
    - Prioritize as Critical – Business Impacting, High - Important, Med – Needed, but workaround, Low – Nice to have
    - Assign to a sprint/release
    - Assign to a developer
  - Change Management
    - Ability to manage issues, assign to a release, and track or chart burn down rate
  - QA/Testing
    - Ability to associate tests to releases
    - Conduct tests that are repeatable, and log success or failure
    - Tests would cover sunny day scenarios, edge cases, and end-to-end
  - Documentation
    - Ability to log and track documentation related to the features, issues, and enhancements
    - Wiki to hold information related to the environments
      - Development: playground developers
      - Testing: Area for the QA Team to test the application
      - Staging: Area to test the build process, allow business to sign off on changes
      - Production
- Configuration Management
  - Deployment Configurations
    - Development
    - Testing
    - Staging
    - Production
  - Source Control
    - Common repository for storing the trunk, branches, and tags
  - Build Server / Continuous Integration
    - Monitors SVN, on change, pulls code, creates a build, and deploys to Test Server
  - Code Analyzer
    - Reviews code against standards

INFRASTRUCTURE MAINTENANCE

Application Administration

- Sizing
  - Adding resources as the users grow
  - Increase/decrease servers in farm/cluster
  - Increase/decrease DB data space
  - Increase/decrease RAM
- Speed
  - Increase or decrease the CPUs speed /
- Hardware
  - Non-issue if hosted in the cloud
  - Will need to

Database Administration

- Handled by the Database Team
- Tuning of the tables/indices

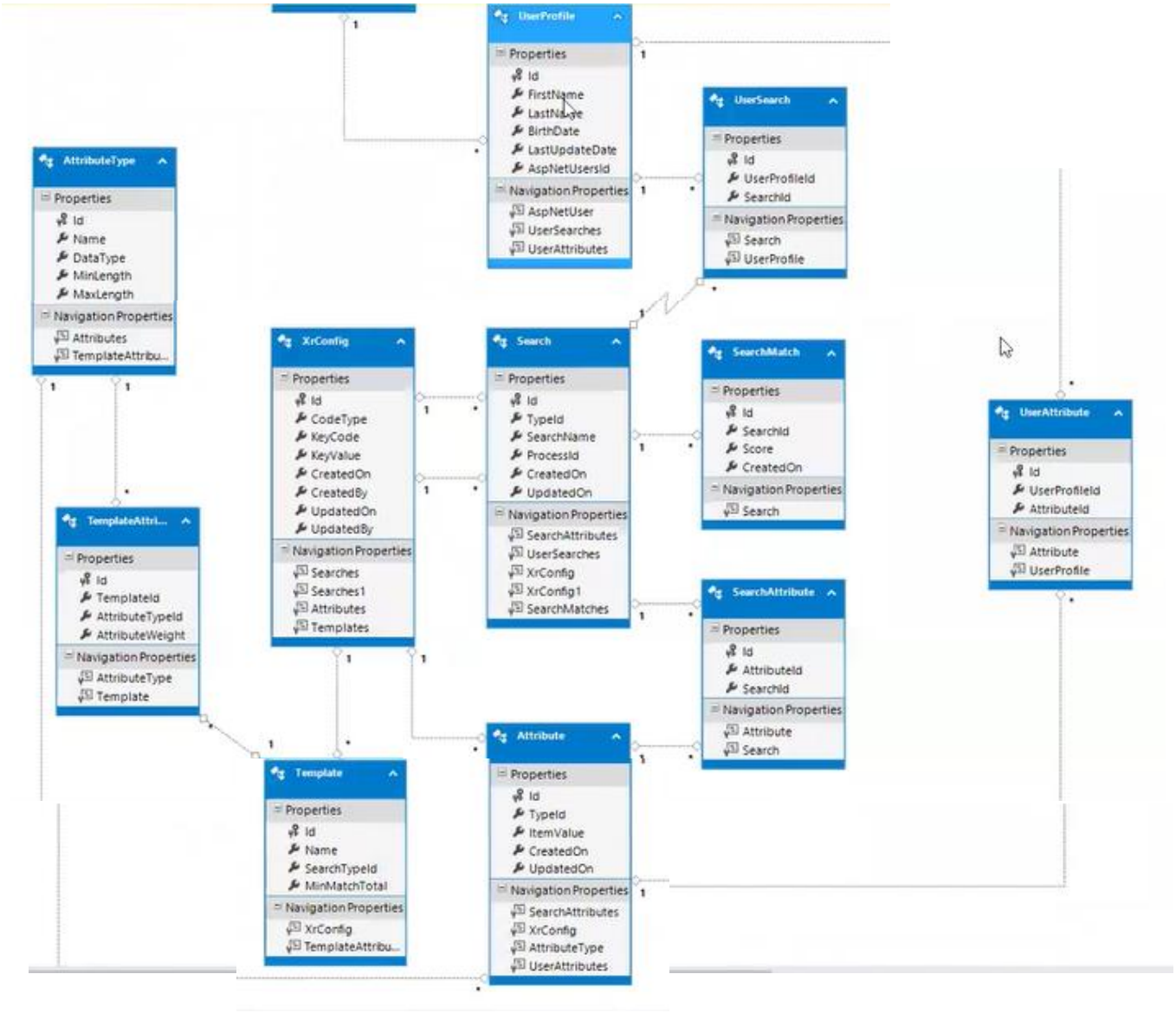
Internal Development Servers

- Source Control Server
  - Hosts SVN
- Project Management Server
  - Hosts Redmine
- SMTP Server
- IDEs for Developers
  - Eclipse
  - Jenkins (Continuous Integration)
  - Lint (Code Review)
- Monitoring software
  - Performance
    - View load
    - View users and activity
  - Exceptions
    - Alerts

SYSTEM DEPLOYMENT TO THE END USER

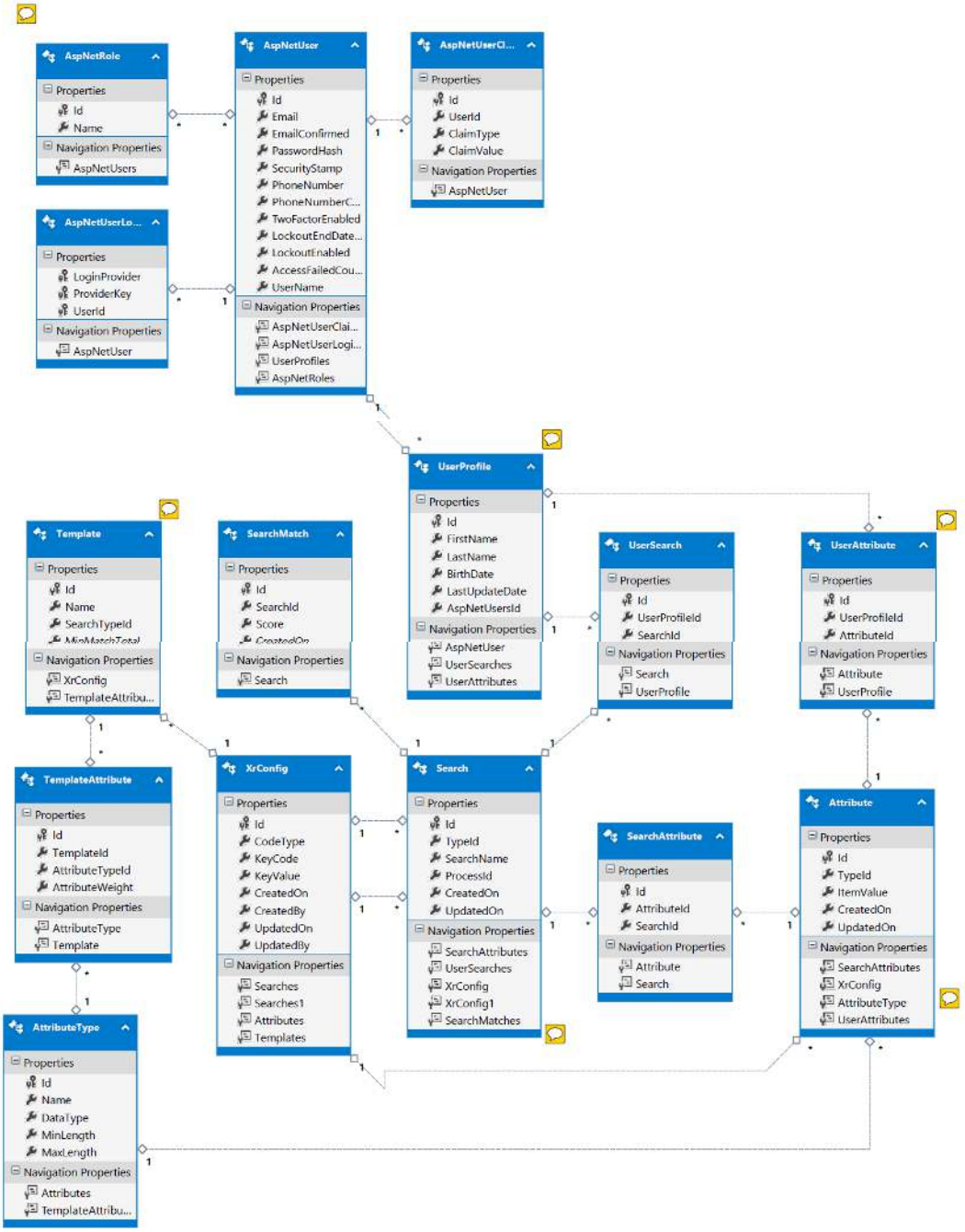
- We are building a prototype; therefore, we are not actually deploying the system to the end user.
- Next year's class, or a graduate student for a Master's Project, will be launching the project through integration of the existing platform with cloud infrastructure, such as Amazon AWS or Azure.

Partially Working Demo Example (Week 5)



Multiple screenshots patched together showing most of [redacted]'s computer screen with his Azure account, showing the Database Structure that he has created with the Demo he has created on his own.

Data Model (Week 6)



Notes and Updates (Week 9)

[redacted] has been working on distance vectors. [redacted] and [redacted] have been working on database calls inside C#. [redacted] has been working on getting the project all setup for everyone to work on. [redacted] has been working on doing some testing.

There have been some updates, but not much – to the Wiki page for Software Group.

[redacted] made this week's full class meeting yesterday on 4/4/2017, while everyone else did not.

[redacted] was on for a minute or so, but he had to get off.