CLIENT MEETING NOTE

JUNE 7TH, 2021
Attendees: Harman Rai, Nicholas Krone, Jacki Zhong, Zeyu Zhang
Time: 6:00 PM

CRM - Customer Relationship Management

Non-negotiable:

- client-server / over IP
    - CRM server
        - Process CRM operations and provide results of operations to application user
    - CRM client
        - Privilege-based access to system
    - Different user levels
        - admin - Can do anything
        - user - Can't change things but can access specific data
        - manager - Import/change records
    - Data storage
        - Not the core element of the project (can be excessive/hectic to work on)
        - Roll own database (design own database - don't replace SQL, basic stuff), cannot be pre-rolled

---

JUNE 9TH, 2021
Attendees: Harman Rai, Nicholas Krone, Jacki Zhong, Zeyu Zhang
Time: 6:00 PM

What it does / not does

- Build a platform that can be used to reach out and maintain relationships with customers
- Many users to use organization to manage different types of relationships and clients
- Lots of records for lots of clients
    - Record example: Fname, Lname, address, phone number
    - Organization records
    - Client records
    - User information/records
    - Customer records are not associative or bound by any sales person
        - Customer chris smith is not defined by the sales person

- Sales people
- Create groups of customers
    - Grouping does not affect anything - just a custom way to order groups/collections of customers
        - Salesmen A has “favorites” group that
- username/password
    - Look into message passing example (objectstream week 2 code example) session based
    - Server needs to keep track of session lifetime (session active?) keep basic

CRM APPLICATION - FOCUSED ON THE CUSTOMERS AND USERS.

Question: What information should the customer/user contain?

SERVER:

- Keep it simple and not devote too much time.
- Back-end server processing, which stores the CRM data.
    - **Records**, which holds
        - Company
        - Users
        - Sales

CLIENT:

- The interaction is done using a JAVA GUI. (Not the web-form for which we're expecting).
    - **Login** to system
        - Check customers records
            - Contacts
            - Notes
- Client should be able to modify their information
    - Like what?
    - Remove customers…
    - Notes
- Privileged permission for users?
    - Able to read and modify all records
    - Data Migration
    - ?

JUNE 14TH, 2021

Attendees: Harman Rai, Nicholas Krone, Jacki Zhong, Zeyu Zhang

Time: 6:00 PM

- User logs in (username and password)
- Empty CRM
    - So add users (sales people - Sales)
    - Add customers
    - Adds customer records, delete customer records, edit
    - Search records
- Sales adds customer records (CRUD)
    - Search record
    - Give reports (based on timestamps - least contacted, most contacted, most engagements - frequency of contact)
    - Calendar (Goodie)
    - Call (Goodie)
    - Group customers
    - Import / export
- 3 levels of users (so far) Admin (IT level, Not a user), Manager, User
    - Admins creates users
    - Managers have Users under them, can add new associates
    - Sales person has records, group records, access records, cannot touch other user accounts/groups
- Persistence (saving everything)
    - Custom storage implementation

<u>Questions:</u>

- **Would users have the ability to login themselves?**
- **How much information should we expect each user to have?**
- **What should the customer be able to do?**
- **How do we send info?**
- **Authentication for each user?**

Visually

SALESFORCE:
COMPANY: CSUEB
USERS: XYZ
SALES: SMITH
GROUP OF SMITHS CUSTOMERS
SALES: GEORGE
GROUP OF GEORGES CUSTOMERS

Note: USER LOGIN -> XYZ PRIVLAGES BY XYZ COMPANY

CRM IS DEPLOYED: CUSTOMER RECORDS.
ORIENTED TOWARDS SALES.

SHUT DOWN THE APPLICATION DOWN AND HAVE UP-TO DATE DATA.
[DVD APPLICATION EXAMPLE]

RECORD -> TOSTRING -> INTO FILE
FILE WOULD HAVE AN ORGANIZED TABLE…
SAVING TO DISK ALL RECORDS TO FILE...

(COMPANY) ADMIN USER (IT PERHAPS):

- LOGIN.
- CREATE USERS LOGINS.
- ADD USERS TO RECORDS.
- REMOVE USER RECORDS.
- RESET USERNAME AND PASSWORD
- SEARCH
- EDIT USERS

MANAGER USER (HAS EMPLOYEES):

- CUSTOMERS
- ADD SALES RECORDS

SALES USER:

- IMPORT/EXPORT THEIR OWN GROUPINGS.
- ACCESS.
- ADD CUSTOMERS.
- SEARCH.

GROUPING:

- CATEGORIES
- SALES

PURPOSE BUT DON'T IMPLEMENT:

- BE ABLE TO PRODUCE REPORTS.
- CALENDAR
- PHONE CALL

JUNE 21ST, 2021
Attendees: Harman Rai, Nicholas Krone, Jacki Zhong, Zeyu Zhang
Time: 6:00 PM

Start to think deeper about the connection between PEOPLE.
How do they interact with each other, **what** do I expect from them?

- If we start to question their purpose, we should update our requirements document.

Think about what kind of messages the classes would send and when.

EXAMPLE : The login interface should have a CONFIRMATION or ERROR message.

Acknowledged, there should be multiple specific types of messages corresponding to the different classes.

Regarding the importance of managing the SERVER.

- Again we should focus on the CRM and keep it simple.
- Basic load configuration, such as uploading a STATIC file for record information.
    - Additional file to maintain the up-to-date information during SHUTDOWN.

Improve on our documentation during client meetings.

JUNE 28th, 2021

Attendees: Harman Rai, Nicholas Krone, Jacki Zhong, Zeyu Zhang
Time: 6:00 PM

Thinking about **how** we should design the GUI.

- Practical approach, do the building and let another department clean it up.
- Developed functionality and the design we are going after.

Show the operations, classes, and the relationships.
We aren't talking to investors, but developers.

Messaging model over the network .

Different types of messages.
Let the thread do everything, approve every

Queue of messages.

Request and then a cancellation.

JULY 7TH, 2021

Attendees: Harman Rai, Nicholas Krone, Jacki Zhong, Zeyu Zhang

Time: 6:00 PM

Presentation Information:

Our project ____

The top 5 requirements ____

Why is it important?

Design elements UML.

Top Requirements.

Top Classes.

Interactions & attributes.

Types of messages.

Demo the design.

Programming:

1. Commenting
2. Documentation.
   (https://www.oracle.com/technical-resources/articles/java/javadoc-tool.html)

JUnit for login testing. Think about white box testing for each method.

JULY 21th, 2021

Attendees: Harman Rai, Nicholas Krone, Jacki Zhong, Zeyu Zhang

Time: 6:00 PM

Design notes:

Implementation should reflect design

Add compile information to design document

Issues:

Zeyu junit testing (debugging issue, works for Nick)