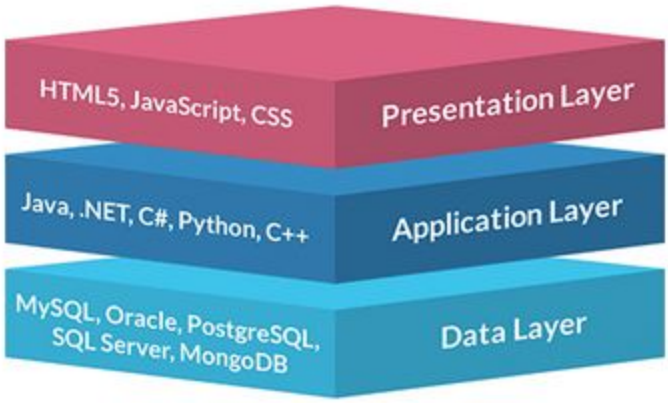
HL design implementation walk through:

What is a three tier architecture?

A 3-tier architecture is a type of software architecture which is composed of three “tiers” or “layers” of logical computing.Doing so gives greater flexibility to development teams by allowing them to update a specific part of an application independently of the other parts. modularizing the user interface, business logic, and data storage layers.

illustration



- **Presentation layer (client)**
  - Provides user interface (the topmost level of the application)
  - Handles the interaction with the user
  - Often referred to as the “front-end”
  - Should not contain Business logic code or data layer code
  - It sends content to browsers in the form of HTML/JS/CSS
  - It communicates with other tiers by which it provides the results to the browser/client side.(we did not show in the HL)

- **Business logic layer (the application layer)**
  - The set of rules for processing information
  - Should not contain Presentation layer or Data layer
  - The logic tier will have the PHP, C++, Python and other programs.
  -

- **Data Layer (data management layer)**
    - Def: manages the physical storage layer, manages access to DB,
    - Often referred to as the “back-end”
    - Should not contain Presentation layer or Business logic code.
    - It provides security, data integrity and support application.
    - The data tier would be some sort of database, such as a MySQL, SQLite or PostgreSQL database.

Pros and Cons of the three tier architecture design:

Advantages	Disadvantages
● Maintainability Each tier is independent of the other tiers, updates or changes can be carried out without affecting the application as a whole. ● Scalability ○ Tiers are based on the deployment of layers, scaling out an application is reasonably straightforward. ○ For example, if you are receiving many web requests but not many requests which affect your application layer, you can scale your web servers without touching your application servers ○ Similarly, if you are receiving many large application requests from only a handful of web users, you can scale out your application an ● Flexibility Because each tier can be managed or scaled independently, flexibility is increased. ● Availability Applications can exploit the modular architecture of enabling systems using easily scalable components, which increases availability. ● Reusability	● Structure is more complex ● Takes a lot of time ● Much complexity to build

Components are reusable ● Faster development ○ division of work web designer does presentation, software engineer does logic, DB admin does data model. ○ specific layer can be upgraded with minimal impact on the other layers iIt can also help improve development efficiency by allowing teams to focus on their core competencies. ● Performance ○ the increased independence created when physically separating different parts of an application minimizes performance issues when a server goes down. ● It reacts to business changes rapidly	

Our implementation of the three tier design architecture:

- **Presentation layer (client)**
  - We left the “views” generic being: “Incoming request”
  - The content in the form of HTML/JS/CSS.
  - And then attempts to pass through the web server which is a part of the presentation.
  -

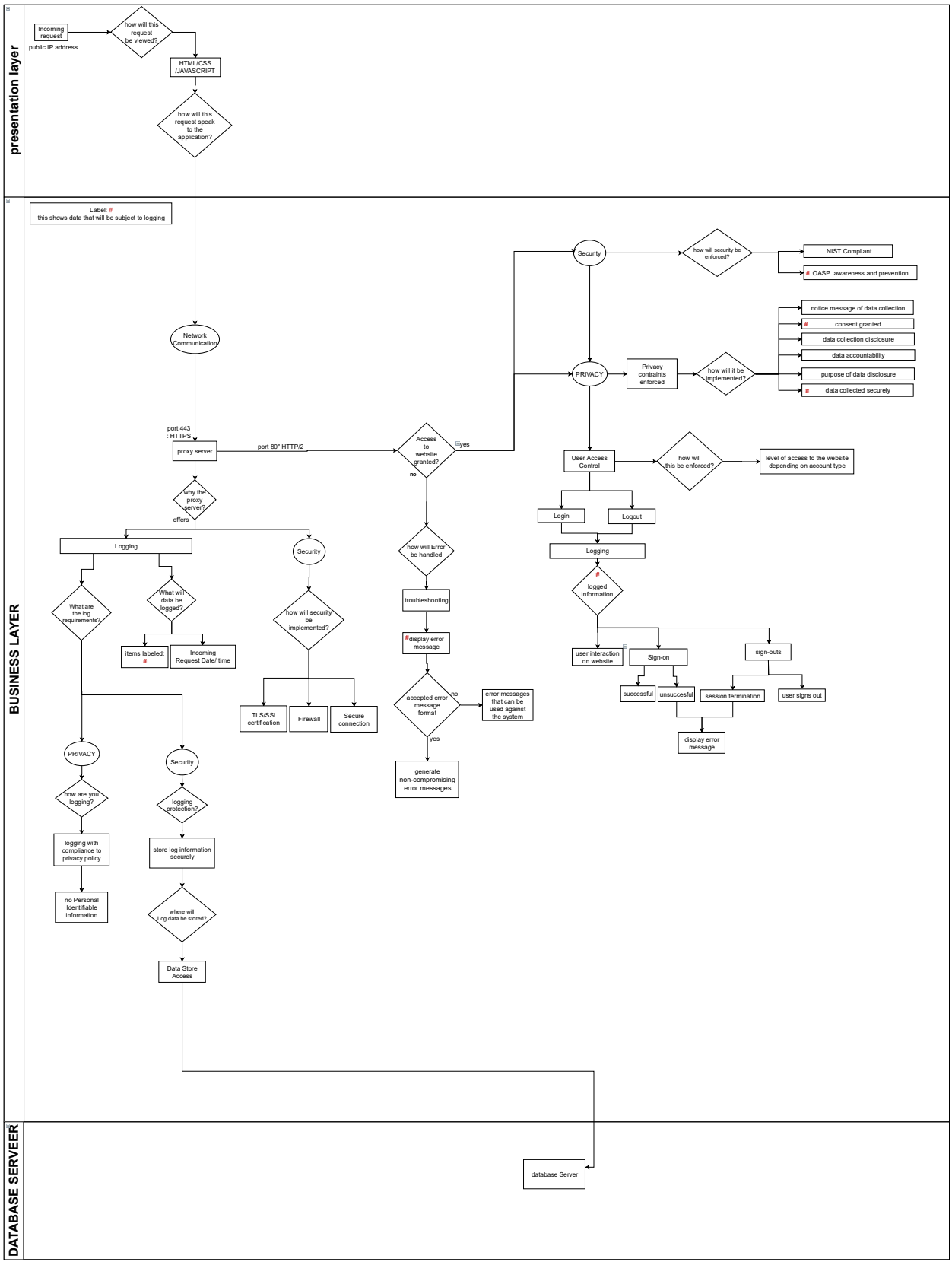
- **Business logic layer (the application layer)**
  - Here you have the following cross cutting core concerns:
    - Security
      - Proxy server
      - NISP compliant
      - OASP awareness to the users
    - Privacy
      - Privacy constraints - listed in the document
      - Secure logging
    - Logging
      - What is being logged.
      - Where it is stored
    - User access control
      - Account permissions
    - Data access store

        - Error handling

- **Data Layer (data management layer)**
    - The database
        - The database storage
        - The database manager

References:

- http://des-megan.blogspot.com/2008/11/advantages-and-disadvantages-of-3-tier.html
- http://www.cs.toronto.edu/~mashiyat/csc309/Lectures/Web%20App%20Architectures.pdf
- https://www.educative.io/blog/how-to-design-a-web-application-software-architecture-101
- https://www.allerin.com/blog/software-product-development
- https://www.ques10.com/p/29000/explain-three-tier-architecture-with-advantages-di/


presentation layer
Incoming request
public IP address
how will this request be viewed?
HTML/CSS /JAVASCRIPT
how will this request speak to the application?
BUSINESS LAYER
Label: #
this shows data that will be subject to logging
Network Communication
port 443 : HTTPS
proxy server
port 80" HTTP/2
why the proxy server?
offers
Logging
What are the log requirements?
What will data be logged?
items labeled: #
Incoming Request Date/ time
PRIVACY
how are you logging?
logging with compliance to privacy policy
no Personal Identifiable information
Security
logging protection?
store log information securely
where will Log data be stored?
Data Store Access
Security
how will security be implemented?
TLS/SSL certification
Firewall
Secure connection
Access to website granted?
yes
no
how will Error be handled
troubleshooting
#display error message
accepted error message format
no
error messages that can be used against the system
yes
generate non-compromising error messages
Security
how will security be enforced?
NIST Compliant
OASP awareness and prevention
PRIVACY
Privacy contraints enforced
how will it be implemented?
notice message of data collection
consent granted
data collection disclosure
data accountability
purpose of data disclosure
data collected securely
User Access Control
how will this be enforced?
level of access to the website depending on account type
Login
Logout
Logging
logged information
user interaction on website
Sign-on
sign-outs
successful
unsuccesful
session termination
user signs out
display error message
DATABASE SERVEER
database Server