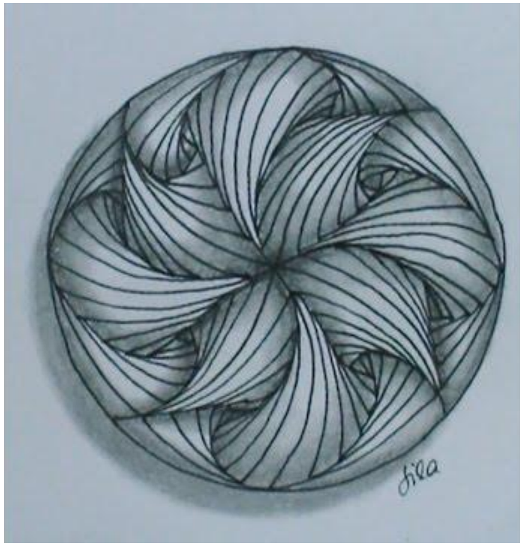
# Paradox Circle



Liliana Aguirre Esparza

Danya Alrawi

Joshua Escareno

Evan Lavelle

Bianca Lujan

# Overview

- Web application with user login and a searchable music and event database.
- Users can add tags to artists which the database keeps track of.
- Other features: Multiple login, Google Calendar integration, Geolocation-tailored results.
- Technologies used: Ruby on Rails, Active Record, PostgreSQL.

# Key Architectural Drivers

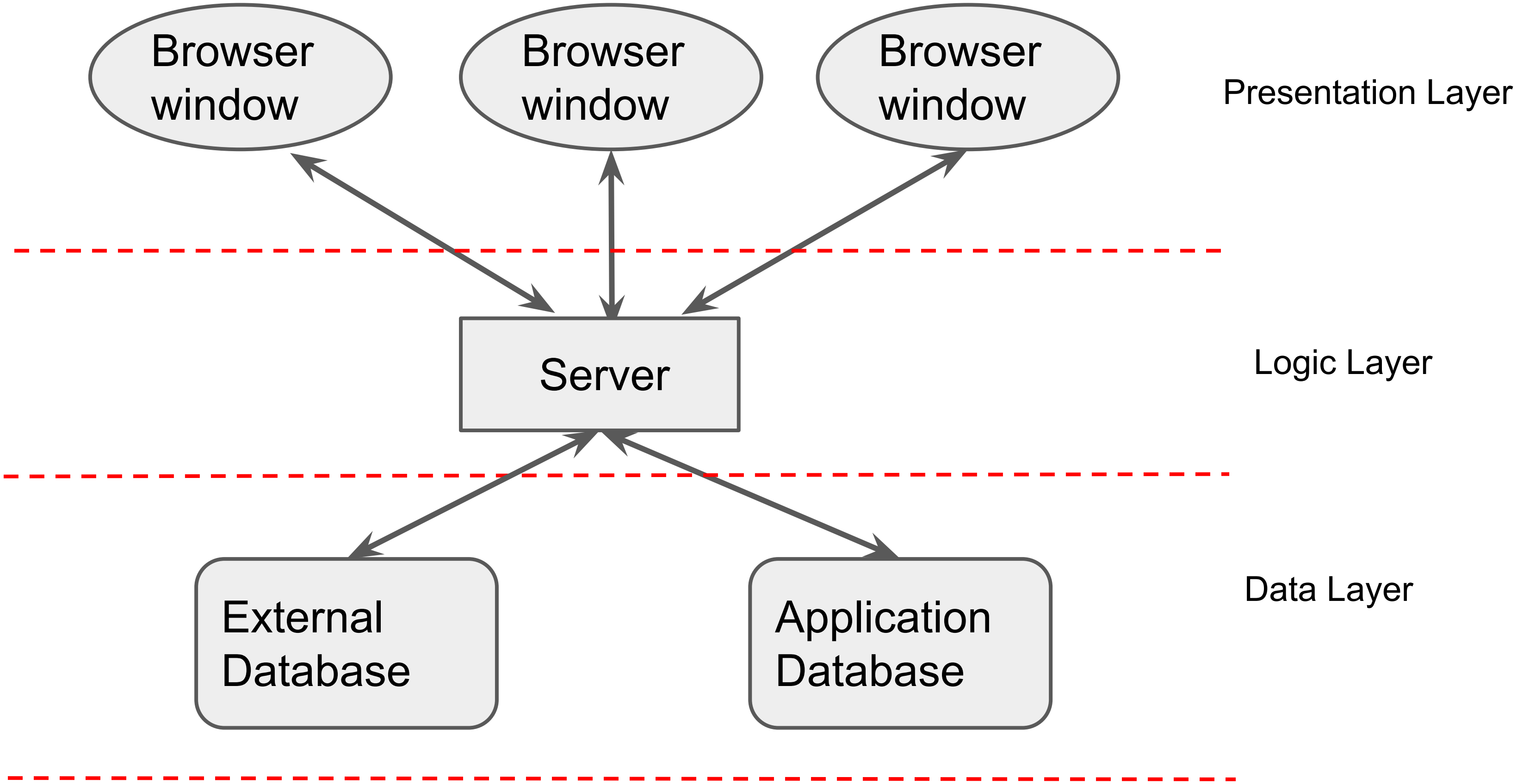
- Users access services by issuing requests to a server through a web browser.
- Data and business logic reside in a server.
- Make use of existing web frameworks (adhere to their design patterns).

# Architectural Style Choices

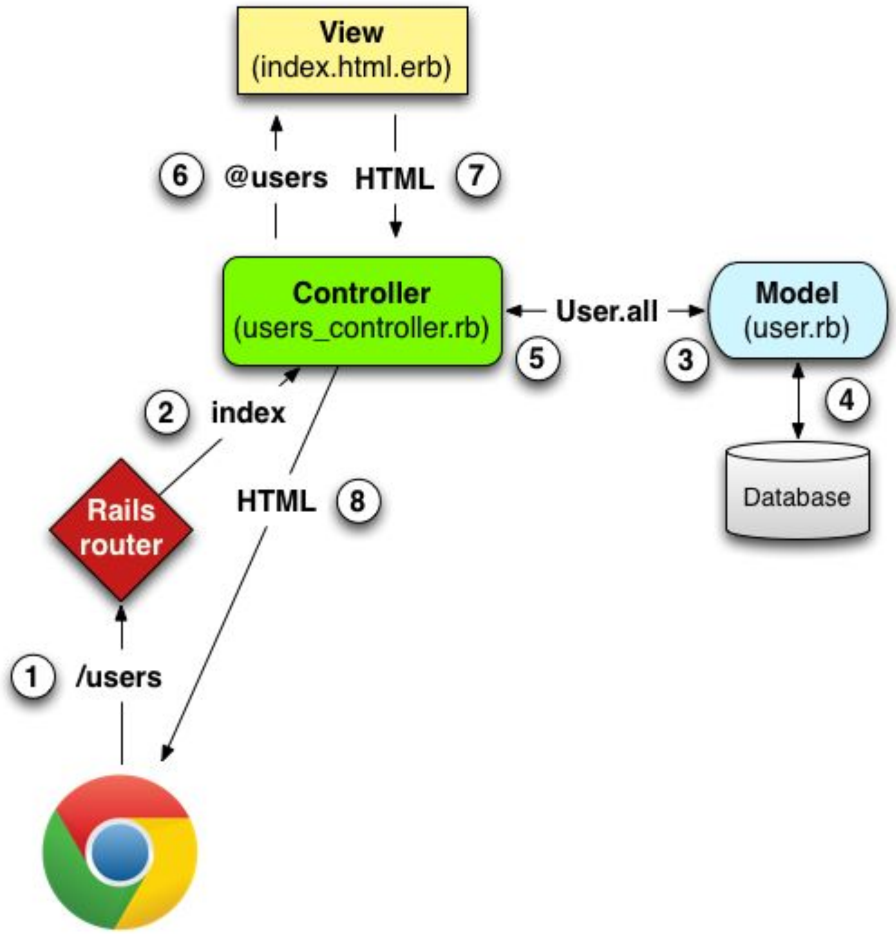
- Peer to Peer: While this is good for scalability and failure tolerance, a decentralized database would not be appropriate for our model and we assume that users don’t wish to host data.

- Client-Server: The standard architecture used for web applications.

# Architecture Diagram



# MVC Pattern: A more detailed view



1. User issues request through browser.
2. Router looks at request, sends it to appropriate controller.
3. Controller checks credentials, if valid request is sent to model.
4. Model performs request, gets data back from database.
5. Model sends this data back to controller.
6. Controller passes the data to view
7. View uses data to create HTML
8. HTML is passed back to the browser by the controller.

Image source: https://softcover.s3.amazonaws.com/636/ruby_on_rails_tutorial_4th_edition/images/figures/mvc_detailed.png

# Conclusion

As a web application, our project lends itself toward a layered client-server architecture. Within this infrastructure, a publish-subscribe architecture may potentially be used for delivery of event notifications, and each request can be looked at as a layered architecture.