System and Software Architecture Description (SSAD)

Spherical Modeling Tool

Team 13

Lyle Franklin, Project Manager

Mehmet Sezer, Requirements Engineer

Guoxiong Xie, Life Cycle Planner

Sait Ilhaner, IIV&V

Rathin Kumar, Operational Concept Engineer

Ari Summer, Builder

Table of Contents

Version History

Date	Author	Version	Changes made	Rationale
12/04/13	LF	1.0	• N/A	• Initial Draft of the SSAD for Team 13
02/18/14	LF	2.0	• Incorporated feedback from client/staff	• Improve document clarity
04/04/14	LF	3.0	• Incorporated feedback from TA	• Add details to several diagrams
04/04/14	RK	3.1	• Review Update	• Review after TRR ARB

A.1 Introduction

A.1.1 Purpose of the SSAD

This document is intended to give an architectural specification of the proposed Spherical Modeling Tool system. The following sections provide both a high-level overview and implementation-level details of the proposed system. Based on our team's current understanding, a system built to this specification will satisfy the client's needs.

A.1.2 Status of the SSAD

This document represents the current architectural body of knowledge our team possesses as of the CCD Presentation. This document will be updated throughout the remainder of the Spring 2014 semester to reflect new architectural knowledge gained during the implementation of this system.

A.2 System Analysis

A.2.1 System Analysis Overview

The Spherical Modeling Tool (SMT) is a data visualization model that uses the shape of a sphere to provide a holistic evaluation of the condition of dynamic systems.

Our mission is to improve productivity, communication, awareness and understanding in a wide variety of areas, including education, healthcare, business, government, and community development by creating a system for information visualization and holistic, integrative thinking.

A.2.1.1 System Context

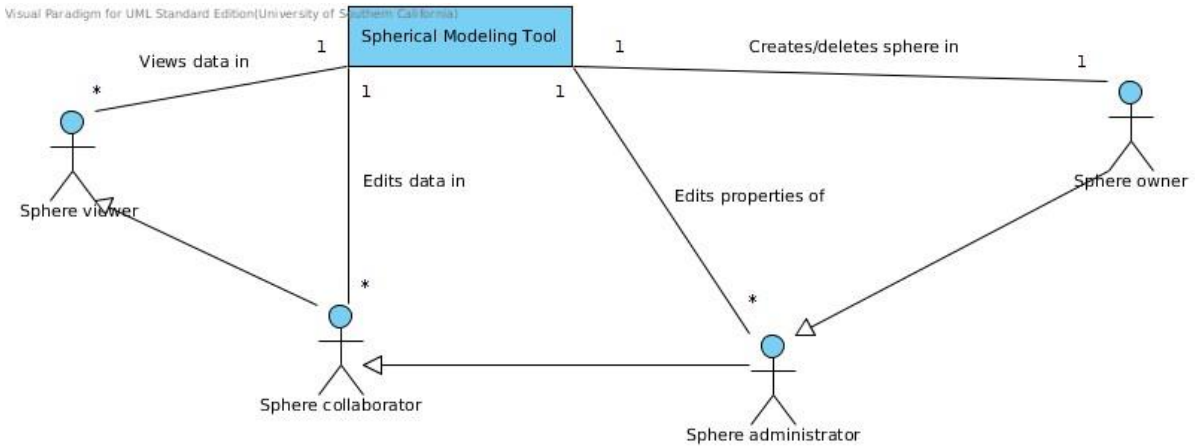

Figure 1: System Context Diagram

Actor	Description	Responsibilities
Sphere owner	Owner of the given sphere	Creator of the sphere, can delete the sphere
Sphere administrator	One of the admins for the given sphere	Can edit or add questions, change title, and grant permissions to other users
Sphere collaborator	A user who was invited to contribute data	Can answer questions for the given sphere
Sphere viewer	A user who was invited to view data	Can view data visualizations and data contributed by other users

A.2.1.2 Artifacts and Information

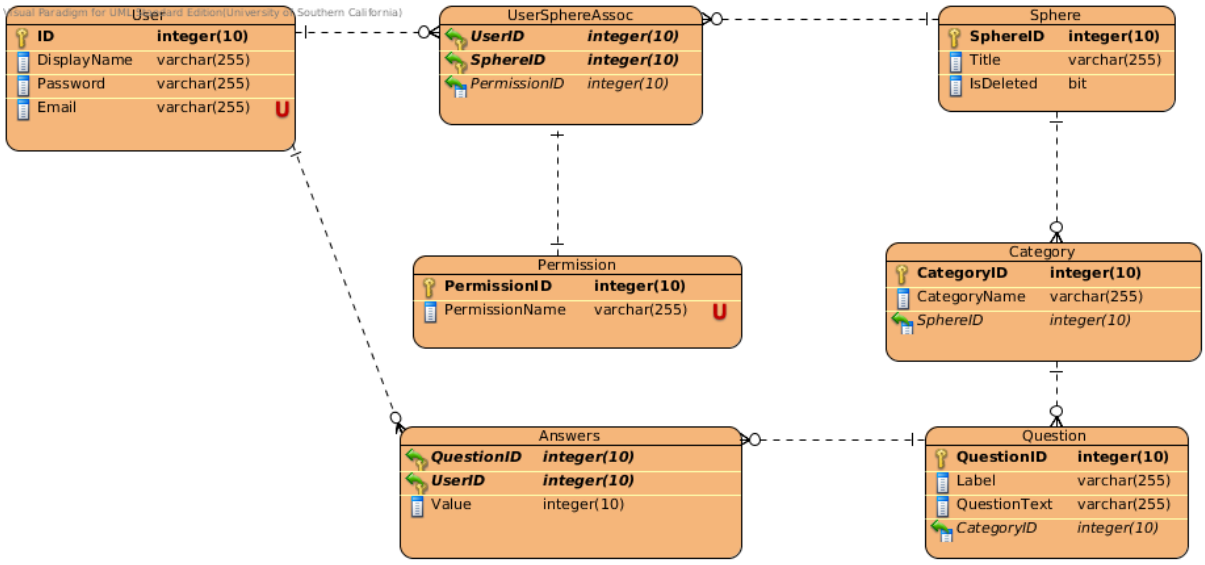

Figure 2: Artifacts & Info. Diagram

Artifact	Purpose
User	Keeps track of account information
UserSphereAssoc	Maintains records of which users have access to which spheres with corresponding levels of permission
Permission	A static table of system permissions (Owner, Admin, etc.)
Sphere	Records of high-level sphere data (e.g. Title)
Category	Belongs to a given sphere; a container for grouping similar questions
Question	A question belonging to a given category
Answer	Keeps track of a user's answer to a given question

A.2.1.3 Behavior

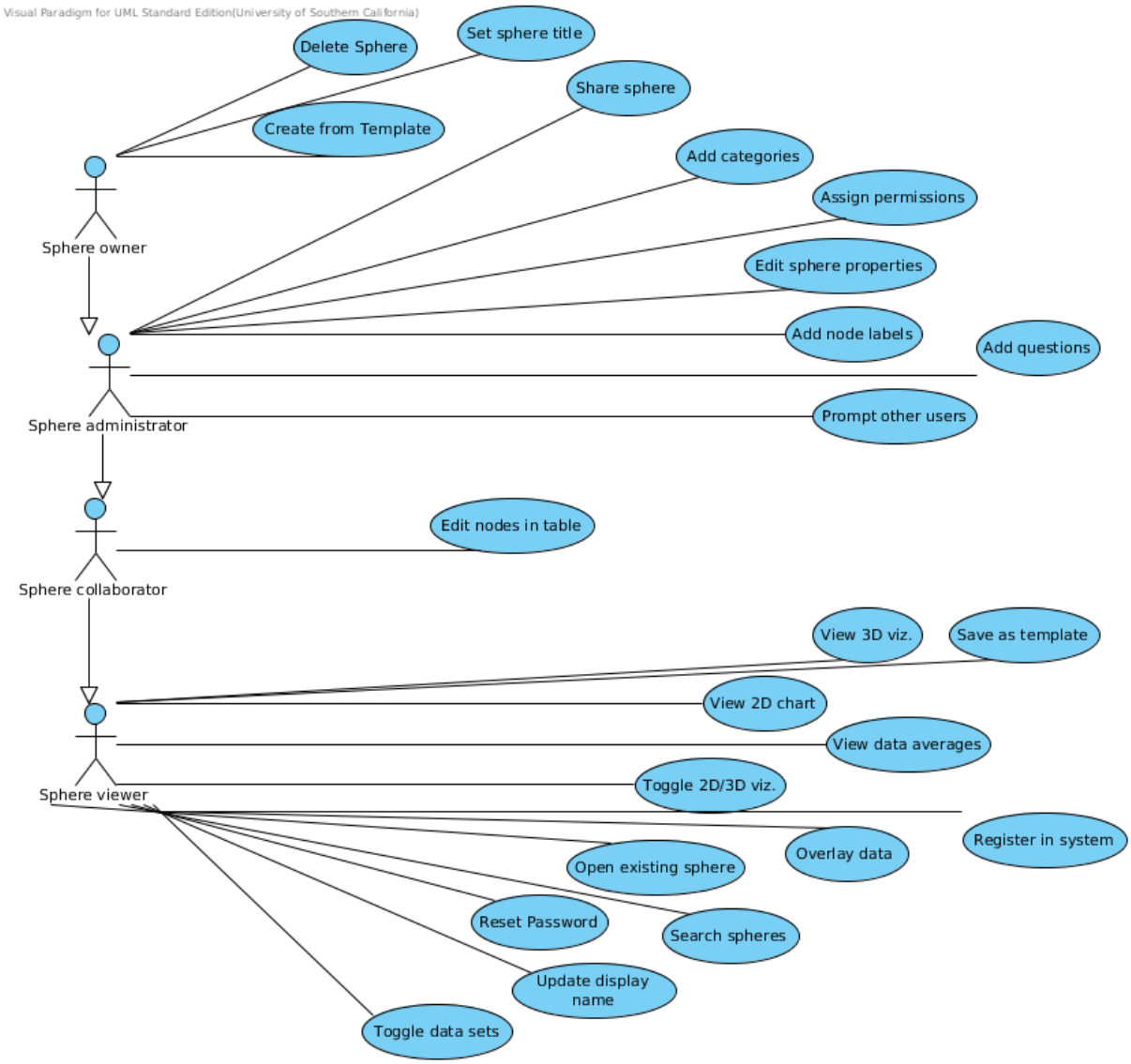

Figure 3: Use Case Diagram

Use Case Descriptions

Create from Template	
Use Case ID	UC28
Brief Description	As a user I can create a sphere from a template.
Actor	Sphere owner
Preconditions	User has access to at least one template. User in on the Create Sphere page.
Post-conditions	A sphere is added to database with questions and categories copied from an existing template, or the create template button is disabled.

Flow of Events		**Actor Input**	**System Response**
	1	User clicks Create from Template.	
	2		Queries database for available templates.
	3		Display a list of templates.
	4	Clicks a template.	
	5		Creates a new sphere in database with copied categories and questions.
	6		Displays the Edit Sphere page.
Alt. Flow		**Actor Input**	**System Response**
	1	User has no templates available.	
	2		Disables Create from template button.

Toggle data sets			
Use Case ID	UC27		
Brief Description	As a user I can select which sets of data show up in the 2D/3D visualizations.\		
Actor	Sphere viewer		
Preconditions	User is on one of the visualization pages.		
Post-conditions	The data set being displayed is changed in 2D or 3D.		
Flow of Events		**Actor Input**	**System Response**
	1	Clicks Filter Data on 2D chart page.	
	2		Queries database for users that have answered questions in this sphere.
	3		Displays list of users

			with a checkbox next to each.
	4	Selects one to many users.	
	5		Updates the chart with only the data for checked users.
Alt. Flow		**Actor Input**	**System Response**
	1	Clicks Filter Data on 3D chart page.	
	2		Queries database for users that have answered questions in this sphere.
	3		Displays list of users with a radio button next to each.
	4	Select one user.	
	5		Updates 3D sphere to show data for selected user.

Search spheres			
Use Case ID	UC26		
Brief Description	As a user I can search my spheres by name and owner.		
Actor	Sphere viewer		
Preconditions	User has access to at least one sphere. User is on Open Sphere page.		
Post-conditions	User is presented with a list of spheres matching search, or No Match Found is displayed.		
Flow of Events		**Actor Input**	**System Response**
	1	User enters a search string.	
	2		Searches database for sphere title or owner that matches search string.

	3		Builds a list of matches and displays them to user.
Alt. Flow		**Actor Input**	**System Response**
	1	User enters a search string.	
	2		Searches database, but finds no matches.
	3		Displays a No Matches Found message to user.

Delete sphere			
Use Case ID	UC25		
Brief Description	As a sphere owner I can delete my spheres.\		
Actor	Sphere owner		
Preconditions	User has Sphere Owner permission. User is on Sphere Edit page.		
Post-conditions	Sphere is logically deleted from database.		
Flow of Events		**Actor Input**	**System Response**
	1	User clicks Delete Sphere.	
	2		Display confirmation prompt.
	3	Clicks Yes.	
	4		Marks sphere as deleted in database.
	5		Redirects user to Home page.
Alt. Flow		**Actor Input**	**System Response**
	1		

Update display name	
Use Case ID	UC24

Brief Description	As a user I can update my display name.\
Actor	Sphere viewer
Preconditions	User has already create an account. User is on Manage Account page.
Post-conditions	Database is updated with new display name, or error message is displayed.

Flow of Events

	Actor Input	System Response
1	Enters new name and clicks submit.	
2		Updates database with new display.
3		Display success message to user.

Alt. Flow

	Actor Input	System Response
1	Enters blank name and clicks submit.	
2		Display error message to user.

Reset password	
Use Case ID	UC23
Brief Description	As a user I can reset my password.\
Actor	Sphere viewer
Preconditions	User has already created account. User is on the login page.
Post-conditions	Database is updated with new password, or reset password link expires.

Flow of Events

	Actor Input	System Response
1	User clicks Forgot Password link.	
2		Prompts for email address.
3	Enters Email and clicks submit.	

	4		Generates security token.
	5		Sends email to user containing reset link and security token.
	6		Displays email send message to user.
	7	Opens email and follows link.	
	8		Checks security token validity.
	9		Prompts user to enter new password.
	10	Enters new password.	
	11		Stores new password in database.
Alt. Flow		**Actor Input**	**System Response**
	1	User clicks reset link after already resetting password.	
	2		System checks token validity.
	3		System responses that link is no longer valid.

Register in system			
Use Case ID	UC22		
Brief Description	As a user I can register so that I can create a sphere. Requires email, name, password.		
Actor	Sphere viewer		
Preconditions	User in not registered. User is on the home page.		
Post-conditions	User is registered, or a form validation error is diplayed.		
Flow of Events		**Actor Input**	**System Response**
	1	User clicks on the “Register” button on the Spherical Modeling Tool	

		page.	
	2		“Registration” page is displayed.
	3	Users fills the “User Name”, “Email Address”, “Create Password” and “Confirm Password” fields and clicks on the submit button.	
	4		If the passwords match (with more than 5 characters) and if it does not exist a user with same user name in the database, a new user data including user name with email address and password is added to the database.
	5		“Open or Create Sphere” page is displayed.
Alt. Flow		**Actor Input**	**System Response**
	1	Users fills the “User Name”, “Email Address”, “Create Password” and “Confirm Password” fields and clicks on the submit button.	
	2		If there exists a mismatch between the passwords, “Registration” page is displayed with the title of “Please Enter The Password again with more than 5 characters or numbers”.
	3		If there exists a user with same user name in the database,

			“Registration” page is displayed with the title of “Someone already has that username. Please try another”.

Add node labels			
Use Case ID	UC21		
Brief Description	As a user I can enter node labels. Weight -> Is my weight where I want it to be? -> Physical		
Actor	Sphere administrator		
Preconditions	User is logged in. User has already entered questions for the nodes. User is on the Edit Sphere page.		
Post-conditions	Nodes are labeled.		
Flow of Events		**Actor Input**	**System Response**
	1	User clicks on the submit button in the “Start a new Sphere” page.	
	2		“Create Sectors and Nodes” page is displayed.
	3	User enters the node names in the “Node name” fields.	
	4		The node names for the sphere is updated on the database.
Alt. Flow		**Actor Input**	**System Response**
	1	User clicks on the “Add Node” button.	
	2		A new “Node Name” field is added to the “Create Sectors and Nodes” page.
	3	User enters the node names in the “Node name” fields.	

	4		The node names for the sphere is updated on the database.

Add categories			
Use Case ID	UC20		
Brief Description	As a user I can enter sectors (categories) for each node. This includes different colored nodes for each sector. This doesn't include rendering effort.		
Actor	Sphere administrator		
Preconditions	User is logged in. User is on the Create Sphere page.		
Post-conditions	Nodes are categorized (colored).		
Flow of Events		**Actor Input**	**System Response**
	1	User clicks on the submit button in the “Start a new Sphere” page.	
	2		“Create Sectors and Nodes” page is displayed.
	3	User enters the question(s) (and node(s)) in the “Question” field(s).	
	4		The question(s) for the sphere is updated on the database.
	5	User enters the Sector Name in the “Sector Label Here” field.	
	6		The Sector Label of the questions of the sphere is updated on the database.
Alt. Flow		**Actor Input**	**System Response**
	1	User clicks on the “Add Sector” button to add more sectors.	

	2		A new “Sector Label Field” is added to the “Create Sectors and Nodes” page.
	3	User enters the Sector Name in the “Sector Label Here” field.	
	4		The Sector Label of the questions of the sphere is updated on the database.

Add questions			
Use Case ID	UC19		
Brief Description	As a user I can generate questions for each node.		
Actor	Sphere administrator		
Preconditions	User is logged in. User is on the Create Sphere page.		
Post-conditions	The questions are added to the database.		
Flow of Events		**Actor Input**	**System Response**
	1	User clicks on the submit button in the “Start a new Sphere” page.	
	2		“Create Sectors and Nodes” page is displayed.
	3	User enters the questions in the fields for nodes.	
	4		The questions for the sphere is updated on the database.
Alt. Flow		**Actor Input**	**System Response**
	1	User clicks on the “Add Node” button to add more nodes for questions.	

	2		New “Question” and “Node Name” fields are added to the “Create Sectors and Nodes” page.
	3	User enters the questions in the fields for nodes.	
	4		The questions for the sphere is updated on the database.

View 3D viz.			
Use Case ID	UC18		
Brief Description	As a user I see the data visualize in a 3D sphere. This includes grouping the nodes by category.		
Actor	Sphere viewer		
Preconditions	User is logged in. User has predefined sphere(s) in database. User has already answered questions. User is on menu page.		
Post-conditions	User is shown 3D visualization.		
Flow of Events		**Actor Input**	**System Response**
	1	User clicks on the “3D Render View” in the “2D Chart view” page.	
	2		The sphere data is taken from the database and according to the 3D visualization algorithm, the sphere is displayed in the “3D Sphere View” page.
Alt. Flow		**Actor Input**	**System Response**
	1		

Set sphere title	
Use Case ID	UC17

<table>
<tr><td>Brief Description</td><td colspan="3">As a user I can title a sphere. No restrictions on duplicate names. Spheres require a unique id.</td></tr>
<tr><td>Actor</td><td colspan="3">Sphere owner</td></tr>
<tr><td>Preconditions</td><td colspan="3">User is logged in. User is on home page.</td></tr>
<tr><td>Post-conditions</td><td colspan="3">A sphere has a title.</td></tr>
<tr><td rowspan="6">Flow of Events</td><td></td><td>Actor Input</td><td>System Response</td></tr>
<tr><td>1</td><td>User clicks on the “Start a new Sphere” button.</td><td></td></tr>
<tr><td>2</td><td></td><td>“Open or Create Sphere” page is displayed.</td></tr>
<tr><td>3</td><td>User writes the title in “Title Your Sphere” field and clicks on the submit button.</td><td></td></tr>
<tr><td>4</td><td></td><td>The title of the sphere is updated in the database.</td></tr>
<tr><td>5</td><td></td><td>“Create Sectors and Nodes” page is displayed.</td></tr>
<tr><td rowspan="2">Alt. Flow</td><td></td><td>Actor Input</td><td>System Response</td></tr>
<tr><td>1</td><td>-</td><td></td></tr>
</table>

<table>
<tr><th colspan="4">Save as template</th></tr>
<tr><td>Use Case ID</td><td colspan="3">UC14</td></tr>
<tr><td>Brief Description</td><td colspan="3">As a user I can save a sphere as a template. All questions and nodes labels exist, but no values.</td></tr>
<tr><td>Actor</td><td colspan="3">Sphere viewer</td></tr>
<tr><td>Preconditions</td><td colspan="3">User is logged in. User is on home page.</td></tr>
<tr><td>Post-conditions</td><td colspan="3">A sphere template without values is created by a user.</td></tr>
<tr><td rowspan="2">Flow of Events</td><td></td><td>Actor Input</td><td>System Response</td></tr>
<tr><td>1</td><td>Clicks on the “Open a Sphere Template”</td><td></td></tr>
</table>

	2		An empty list of questions and nodes displayed.
	3	Enters the questions and nodes.	
	4	Clicks "Save Template"	
	5		Saves template to database.
Alt. Flow		**Actor Input**	**System Response**
	1	-	

Open existing sphere			
Use Case ID	UC13		
Brief Description	As a user I can open an existing sphere project. This include loading a sphere from the database.\		
Actor	Sphere viewer		
Preconditions	User is logged in. Use has already created a sphere. User is on the home page.		
Post-conditions	User successfully opens a sphere, or an empty list is displayed.		
Flow of Events		**Actor Input**	**System Response**
	1	Clicks "Open a Saved Sphere"	
	2		Requests sphere list of the user from the database.
	3		Displays list of saved spheres.
	4	Selects one sphere from the list.	
	5		Displays 2D view of selected sphere.
Alt. Flow		**Actor Input**	**System Response**
	1	Clicks on the “Open a saved sphere” button.	

	2		Requests sphere list of the user from the database.
	3		If there is no saved spheres in the database, the empty list is displayed.

Overlay data			
Use Case ID	UC12		
Brief Description	As a user I can overlay data from different users so that I can compare different shapes (w/different colors). This only includes the 2D rendering.		
Actor	Sphere viewer		
Preconditions	At least one user has entered data for given sphere. User has permission to view sphere. User is on 2D chart page.		
Post-conditions	User is shown multiple data sets overlaid.		
Flow of Events		**Actor Input**	**System Response**
	1	Clicks Compare button.	
	2		Displays list of users that have entered data for this sphere.
	3	Does not specifically select a subset of users.	
	4		Renders a different data set for each user, overlaid on 2D chart.
Alt. Flow		**Actor Input**	**System Response**
	1	Selects a subset of users that have entered data.	
	2		Renders data for only selected users, overlaid on 2D chart.

<table>
<tr><th colspan="4">Edit sphere properties</th></tr>
<tr><td>Use Case ID</td><td colspan="3">UC11</td></tr>
<tr><td>Brief Description</td><td colspan="3">As a user I can edit/add/remove nodes (questions), sectors (categories), and sphere title (with admin permission). Assume sphere creation functions already exist.</td></tr>
<tr><td>Actor</td><td colspan="3">Sphere administrator</td></tr>
<tr><td>Preconditions</td><td colspan="3">User has already created sphere. User has access to modify sphere. User is on sphere edit page.</td></tr>
<tr><td>Post-conditions</td><td colspan="3">DB is updated with new sphere properties, or an error message is displayed.</td></tr>
<tr><td>Flow of Events</td><td></td><td>Actor Input</td><td>System Response</td></tr>
<tr><td></td><td>1</td><td>Clicks Edit sphere.</td><td></td></tr>
<tr><td></td><td>2</td><td></td><td>Displays sphere properties in editable table.</td></tr>
<tr><td></td><td>3</td><td>Changes values in table.</td><td></td></tr>
<tr><td></td><td>4</td><td>Clicks Update.</td><td></td></tr>
<tr><td></td><td>5</td><td></td><td>Validates input.</td></tr>
<tr><td></td><td>6</td><td></td><td>Updates DB with new properties.</td></tr>
<tr><td></td><td>7</td><td></td><td>Display success message to user.</td></tr>
<tr><td>Excep. Flow</td><td></td><td>Actor Input</td><td>System Response</td></tr>
<tr><td></td><td>1</td><td>Enters invalid values in table.</td><td></td></tr>
<tr><td></td><td>2</td><td></td><td>Displays error message to user.</td></tr>
<tr><td></td><td>3</td><td></td><td>Does not update DB with new properties.</td></tr>
</table>

View data averages	
Use Case ID	UC10
Brief Description	As a user I can view the average of all (or subset) of

<table>
<tr><td></td><td colspan="3">selected user data in a single system. This does not include collaborative editing features. This does not include rendering the overlay.</td></tr>
<tr><td>Actor</td><td colspan="3">Sphere viewer</td></tr>
<tr><td>Preconditions</td><td colspan="3">At least one user has entered data for this sphere. User has access to view sphere. User is on the menu page.</td></tr>
<tr><td>Post-conditions</td><td colspan="3">User is shown data averages.</td></tr>
<tr><td>Flow of Events</td><td></td><td>Actor Input</td><td>System Response</td></tr>
<tr><td></td><td>1</td><td>User clicks on Answers.</td><td></td></tr>
<tr><td></td><td>2</td><td></td><td>Display some data for each user that sphere is shared with.</td></tr>
<tr><td></td><td>3</td><td>User does not specifically select any subset of users.</td><td></td></tr>
<tr><td></td><td>4</td><td></td><td>Computes average of all users in sphere.</td></tr>
<tr><td></td><td>5</td><td></td><td>Displays values to user.</td></tr>
<tr><td>Alt. Flow</td><td></td><td>Actor Input</td><td>System Response</td></tr>
<tr><td></td><td>1</td><td>Selects a subset of users in sphere.</td><td></td></tr>
<tr><td></td><td>2</td><td></td><td>Computes average data for given subset.</td></tr>
<tr><td></td><td>3</td><td></td><td>Displays data to user.</td></tr>
</table>

Assign Permissions	
Use Case ID	UC09
Brief Description	As a user I can assign permissions to other registered users. This includes defining (hardcoded) user roles. This doesn't include how those permissions are used/enforced.
Actor	Sphere administrator
Preconditions	User has already shared sphere with other users. User is on the menu page.
Post-conditions	Other user is given a different system permission.

Flow of Events		Actor Input	System Response
	1	Clicks Share button.	
	2		Displays list of users with access to sphere. Each user has permission dropdown
	3	Changes permission for one or more users.	
	4	Clicks Update button.	
	5		Updates DB with new permission info.
	6		Display success message to user.
Alt. Flow		**Actor Input**	**System Response**
	1	**Clicks X button next to previously shared user.**	
	2		**Removes permission from user in database.**
	3		Deletes user response data from sphere if the user had answered questions.
	4		Display Success Message.

Toggle 2D/3D viz.	
Use Case ID	UC08
Brief Description	As a user I can toggle between 2D and 3D visualizations. This task only concerns switching between views, not implementing those views.
Actor	Sphere viewer
Preconditions	User is on 2D or 3D page. User has access to view sphere.
Post-conditions	User is shown the 2D visualization, or the 3D visualization.

Flow of Events		**Actor Input**	**System Response**
	1	User is on 2D page.	
	2	Clicks 3D button.	
	3		Displays 3D visualization page.
Alt. Flow		**Actor Input**	**System Response**
	1	User is on 3D page.	
	2	Clicks 2D button.	
	3		Displays 2D visualization page.

Prompt other users			
Use Case ID	UC06		
Brief Description	As a user I can prompt other users to enter data (notification through app and email).		
Actor	Sphere administrator		
Preconditions	First user has already told system to Share sphere with second user. Both users have accounts in the system. User is on the share page.		
Post-conditions	Second users receives email and in-app notification.		
Flow of Events		**Actor Input**	**System Response**
	1	Users tells system to share sphere with another user.	
	2		Sends email notification to second user.
	3		Adds notification record in DB.
	4	Second user logs into system.	
	5		Displays in-app notification to user.
	6	User clicks notification.	
	7		Redirects to sphere edit

			page.
Alt. Flow		**Actor Input**	**System Response**
	1	Second user logs into system.	
	2		Displays in-app notification to user.
	3	User does not click notification.	
	4	User logs out.	
	5		Logs out user.
	6	User logs back in.	
	7		Logs in user.
	8		Displays notification again.

Share sphere			
Use Case ID	UC05		
Brief Description	As a user I can share a sphere with another user. This includes allowing another user to view your sphere and edit data (based on permissions).		
Actor	Sphere administrator		
Preconditions	User has created a sphere. User knows email address of second user. Second user has already created an account in the system. User is on the menu page.		
Post-conditions	Second user can view/edit shared sphere, or an error message is displayed.		
Flow of Events		**Actor Input**	**System Response**
	1	First user clicks Share.	
	2		Displays list of users that sphere is already shared with.
	3	Enters email of new user.	
	4	Clicks Add user.	
	5		Adds association in DB.

	6		Sends notification to second user.
	7		Displays success message to first user.
Excep. Flow		**Actor Input**	**System Response**
	1	Enters email of user not in system.	
	2		Displays error message to first user.

Edit nodes in table			
Use Case ID	UC03		
Brief Description	As a user I can edit data in table format (editing node values).		
Actor	Sphere collaborator		
Preconditions	User is on sphere edit page. User has permission to edit sphere.		
Post-conditions	DB is updated with new values, or Update button is disabled.		
Flow of Events		**Actor Input**	**System Response**
	1	Changes values of one or more nodes.	
	2	Clicks Update.	
	3		Updates DB with new values.
	4		Redirects to sphere overview.
Alt. Flow		**Actor Input**	**System Response**
	1	Changes no values.	
	2	Clicks Update.	
	3		Button is grayed out until values are changed. DB is not updated.

<table>
<tr><th colspan="4">View 2D chart</th></tr>
<tr><td>Use Case ID</td><td colspan="3">UC01</td></tr>
<tr><td>Brief Description</td><td colspan="3">As a user I can visualize the data in a 2D chart. This includes grouping the nodes into categories.</td></tr>
<tr><td>Actor</td><td colspan="3">Sphere viewer</td></tr>
<tr><td>Preconditions</td><td colspan="3">User has logged in, user is on sphere list page. User has already answered questions.</td></tr>
<tr><td>Post-conditions</td><td colspan="3">User is presented with a 2D chart, or an error message is displayed.</td></tr>
<tr><td rowspan="4">Flow of Events</td><td></td><td>Actor Input</td><td>System Response</td></tr>
<tr><td>1</td><td>Selects a sphere from a list of accessible spheres</td><td></td></tr>
<tr><td>2</td><td></td><td>Queries server for latest data</td></tr>
<tr><td>3</td><td></td><td>Renders 2D polar chart using data from server</td></tr>
<tr><td rowspan="4">Alt. Flow</td><td></td><td>Actor Input</td><td>System Response</td></tr>
<tr><td>1</td><td>Selects a sphere that has no associated data</td><td></td></tr>
<tr><td>2</td><td></td><td>Queries server for data</td></tr>
<tr><td>3</td><td></td><td>Display textual message to user indicating sphere has no data</td></tr>
</table>

A.2.2 System Analysis Rationale

Some aspects of the system described thus far may be misunderstood. Here are several extended explanations of features that could be misinterpreted:

- User permission levels (Owner, admin, collaborator, viewer) described previously are with respect to a given sphere. For example, Tom could be the Owner of the Health Sphere, a Viewer of the Community Sphere, and have no access to the Project Sphere. Any user can create any number of spheres, but that user must be granted permission to access another user's sphere(s).
- When a user is added as a Collaborator to a sphere, the user must answer the sphere's questions before viewing any data.
- Anyone with access to the sphere can see data from all the users that have answered questions for that sphere.
- Questions must be answerable with an integer value from 1 to 9. For example, “How happy are you with your weight?”.

A.3 System Design

A.3.1 Design Overview

A.3.1.1 System Structure

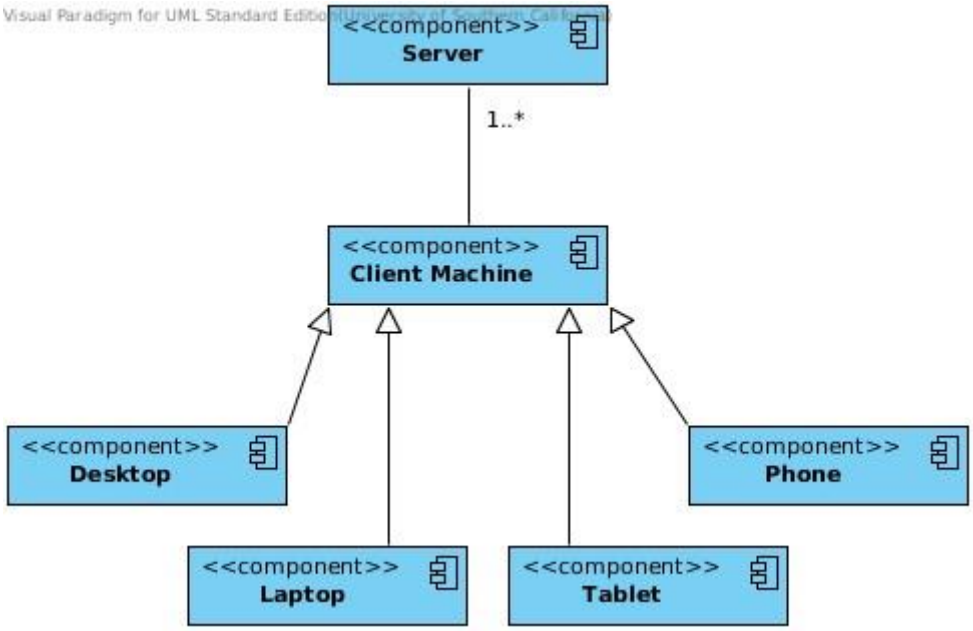

Figure 4: Hardware Component Diagram

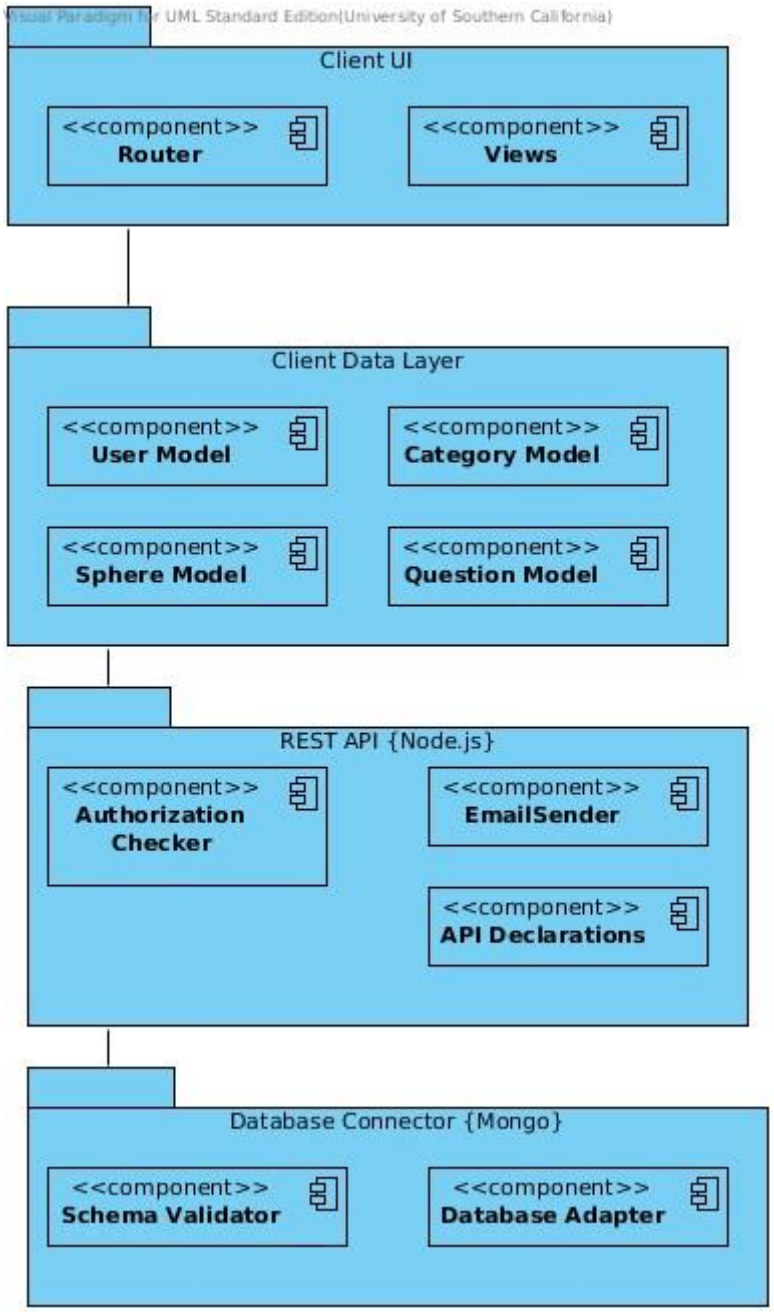

Figure 5: Software Component Diagram

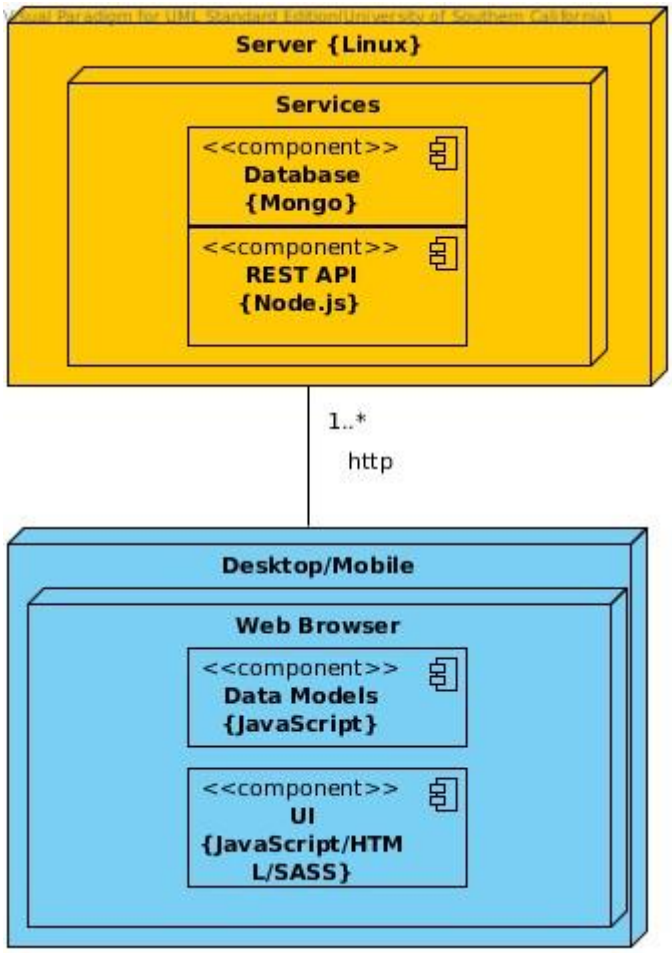

Figure 6: Deployment Diagram

A.3.1.2 Design Classes

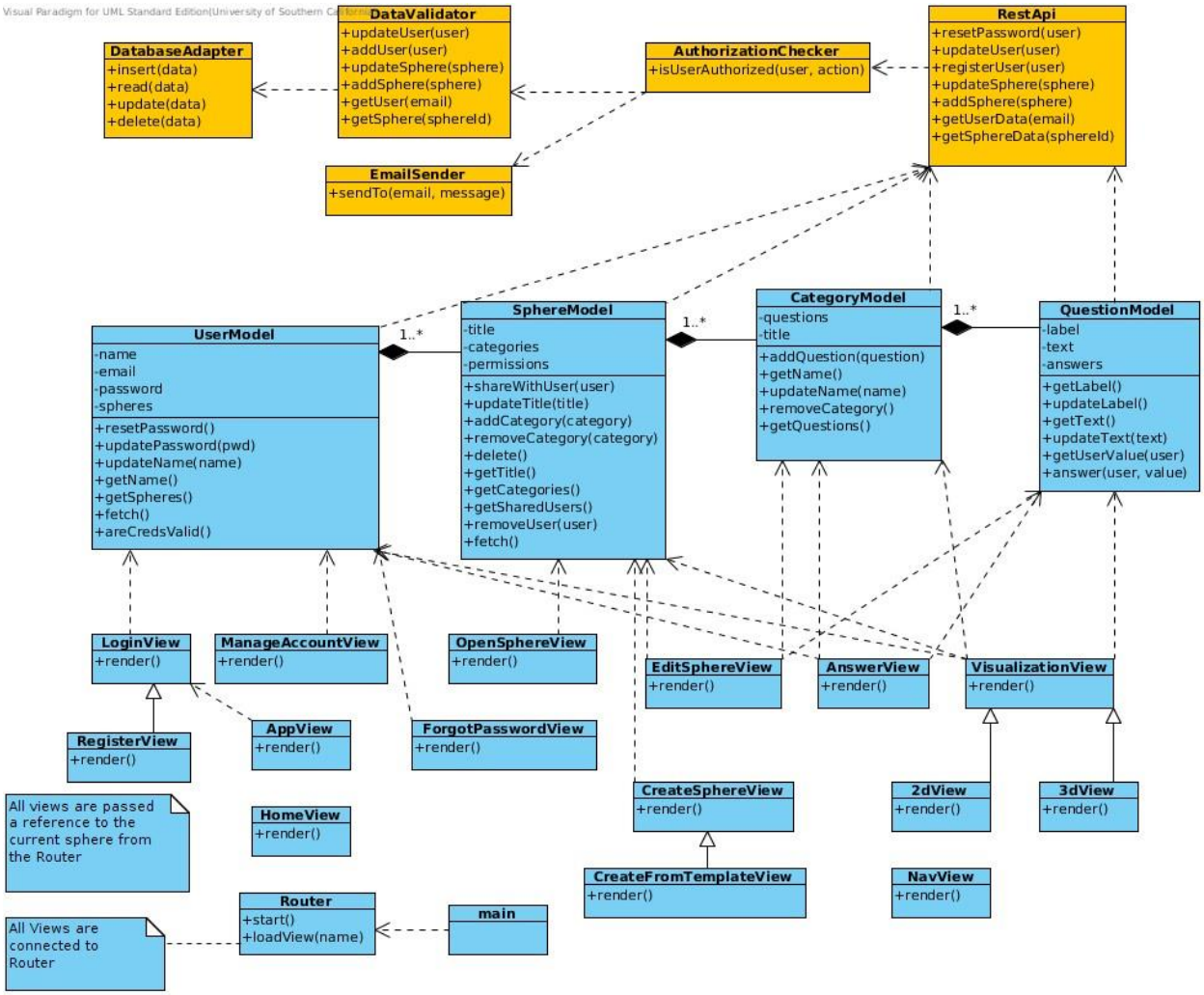

Figure 7: Class Diagram

A.3.1.3 Process Realization

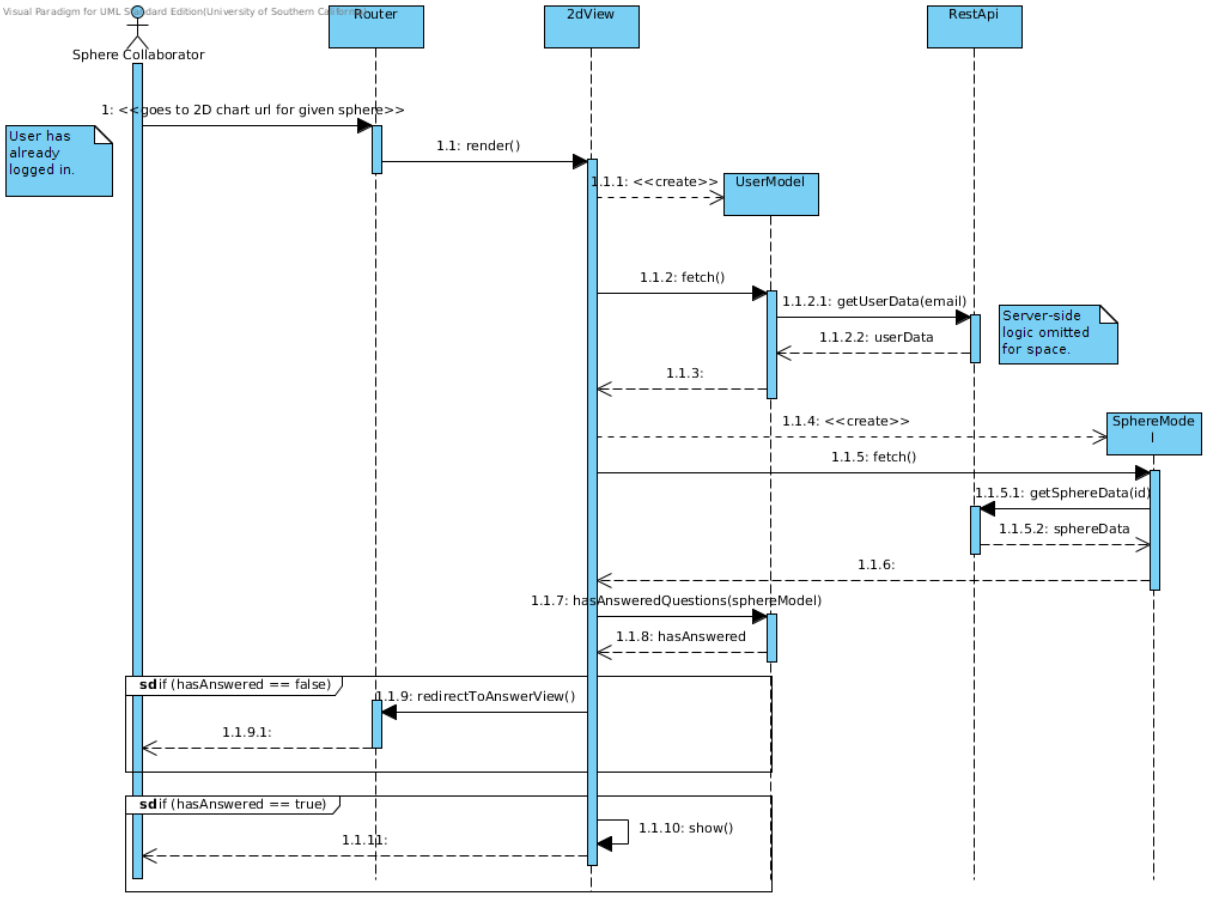

Figure 8: Collaborator Sequence Diagram

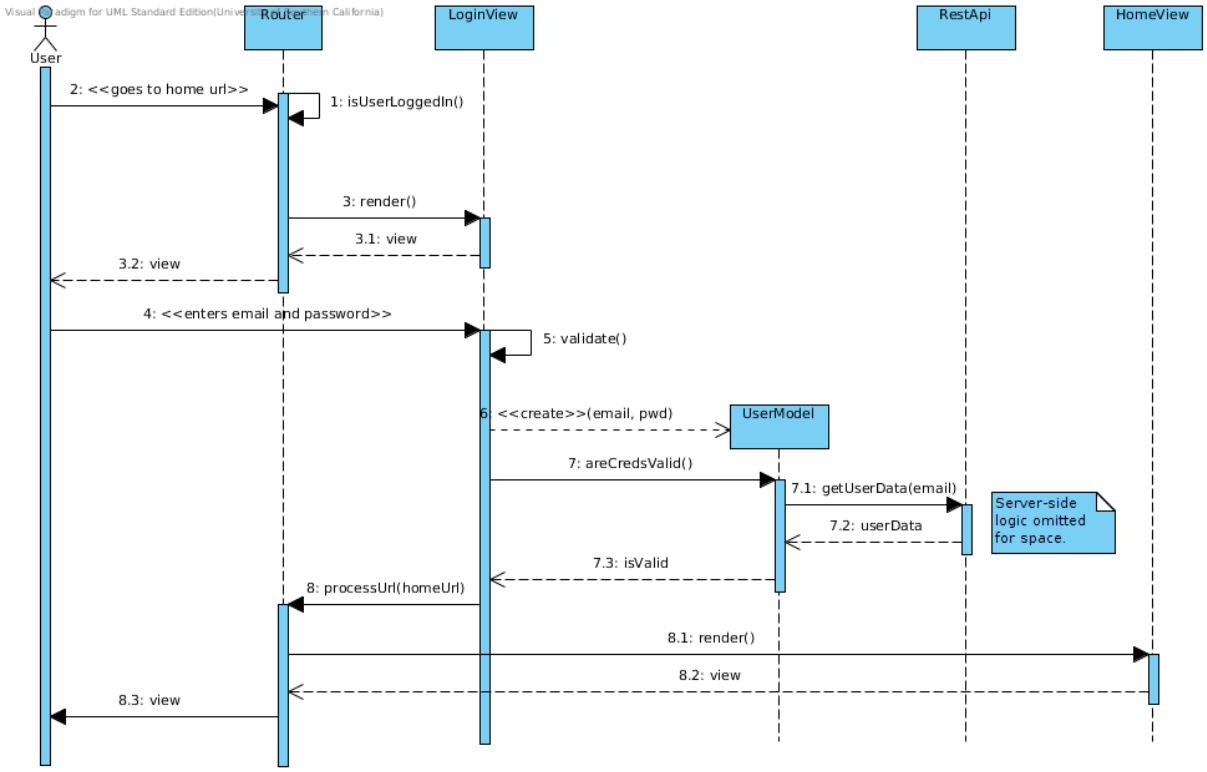

Figure 9: Login Sequence Diagram

A.3.2 Design Rationale

The Hardware Component, Software Component, and Deployment Diagrams show that this system is a web-application which utilizes a client-server architecture. The Hardware Component Diagram also shows this application will be useable on both desktop and mobile devices.

The Class Diagram shows an implementation-level view of the system. In order to achieve the speed and fluidity of a native application, most of the logic will take place client-side using JavaScript. The Class Diagram shows server-side components in orange, and the client-side components in blue.

The server-side will be a lightweight REST API which is primarily concerned with data validation and storage, as well as user permissions.

The client-side will follow a Model-View pattern favored by many JavaScript frameworks. A variety of models will be constructed by fetching data from the server, and dynamic Views will be rendered using these Models. As indicated by a note in the diagram, a Router component will map a given URL to the correct view. User input will update these Models and the changes will propogate back to the server.

The sequence diagrams detail two non-obvious use cases in our system. The first describes a Sphere Collaborator trying to visit the 2D Chart page. The system checks whether the user has already answered questions for the given sphere. If the user has answered the questions, the system will display the 2D chart page, else the user is redirected to the Answer page. The second sequence diagram describes the Login use case. The system validates the user's credentials server-side and only displays the requested page if the credentials match.

A.5 Architectural Styles, Patterns, and Frameworks

Name	Description	Benefits, Limitations
Node.js	Server-side JavaScript framework used to build event-based web applications	Benefits: - Easy to setup and deploy - Application lends itself to event-based style - Native mapping to/from JSON data (Everything is JavaScript) - Free Limitations: - Relatively new technology - Team members not experienced with Node.js
Backbone.js	Client-side JavaScript Model-View framework	Benefits: - Gives structure to client-side code - Client-side code can switch views immediately without a round trip to server - Several team members have experience with Backbone.js - Free Limitations: - It could be argued that Angular.js is becoming a more popular option - Moderate learning curve for team members without experience
MongoDB	A popular NoSQL database that supports documents in a JSON format	Benefits: - JSON format requires minimal data mapping to/from JavaScript - Integrates well with Node.js - Dynamic schema is accomidating of changes in data structure - Well-support by many hosting services - Free Limitations: - Team members lack experience in this technology - We may discover the data lends itself better to a relational database