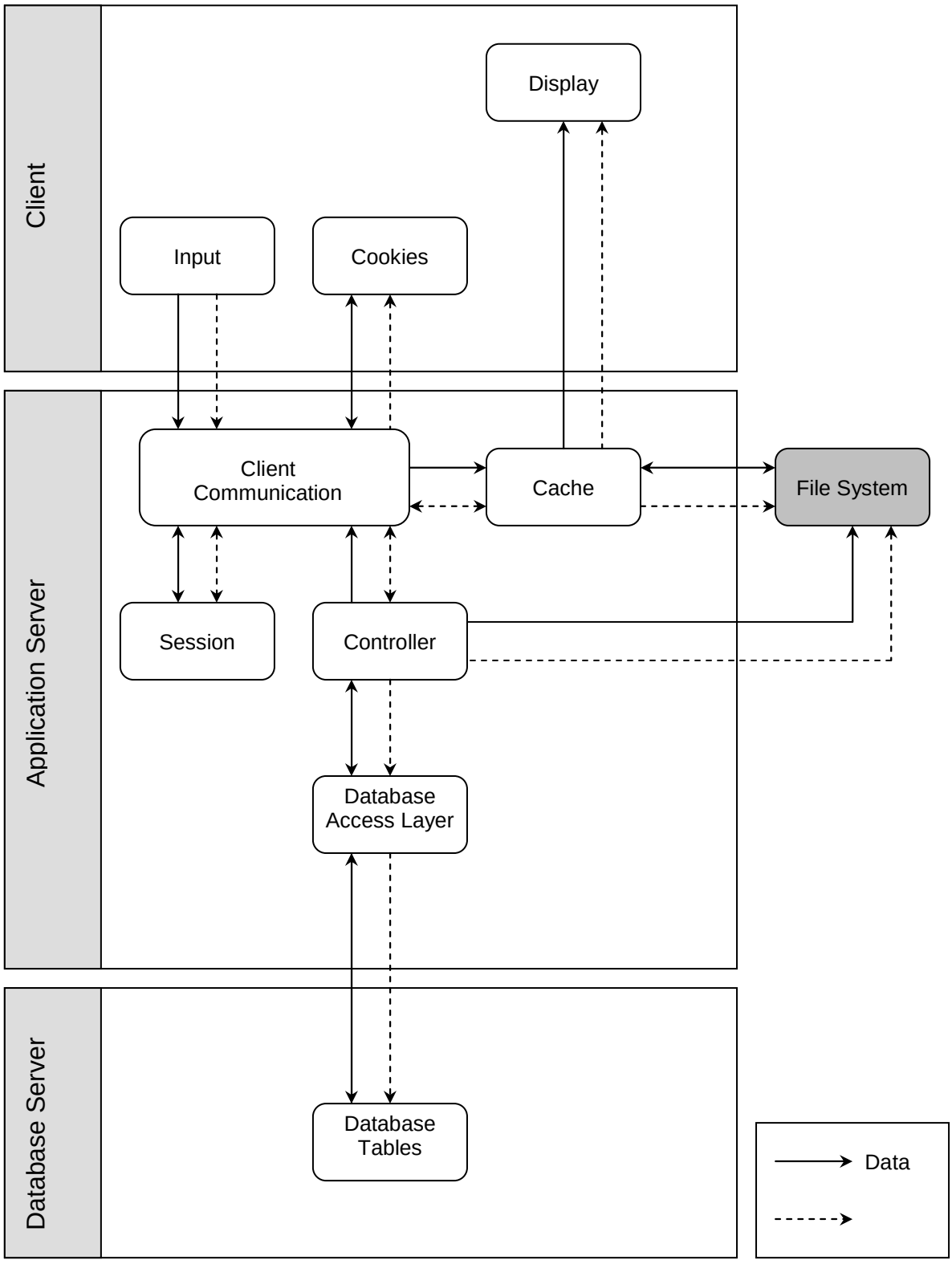
Course Information Management System

Group 2

David Chang
Linda Chowdhury
Oscar Fitinghoff
Patrik Parberg
Tomas Hansson

2.2 Overall Architecture Description

The system uses a three-tier client-server architecture, where the clients are web browsers and the servers consists of an application server and a database server.



The web browsers contain no business logic and manage the user interface. The web browser also stores cookies to enable the application server to identify the user. It also enables the user to send input to the application server.

The application server handles the incoming requests by retrieving and storing data form the database server. The application server uses caching to limit the number of database queries.

The application server consists of mainly three logical layers for processing requests. The first layer (client communication) interprets the user request and calls upon the appropriate controllers. In between the client communication layer and the client lies a cache layer which caches output for some requests.

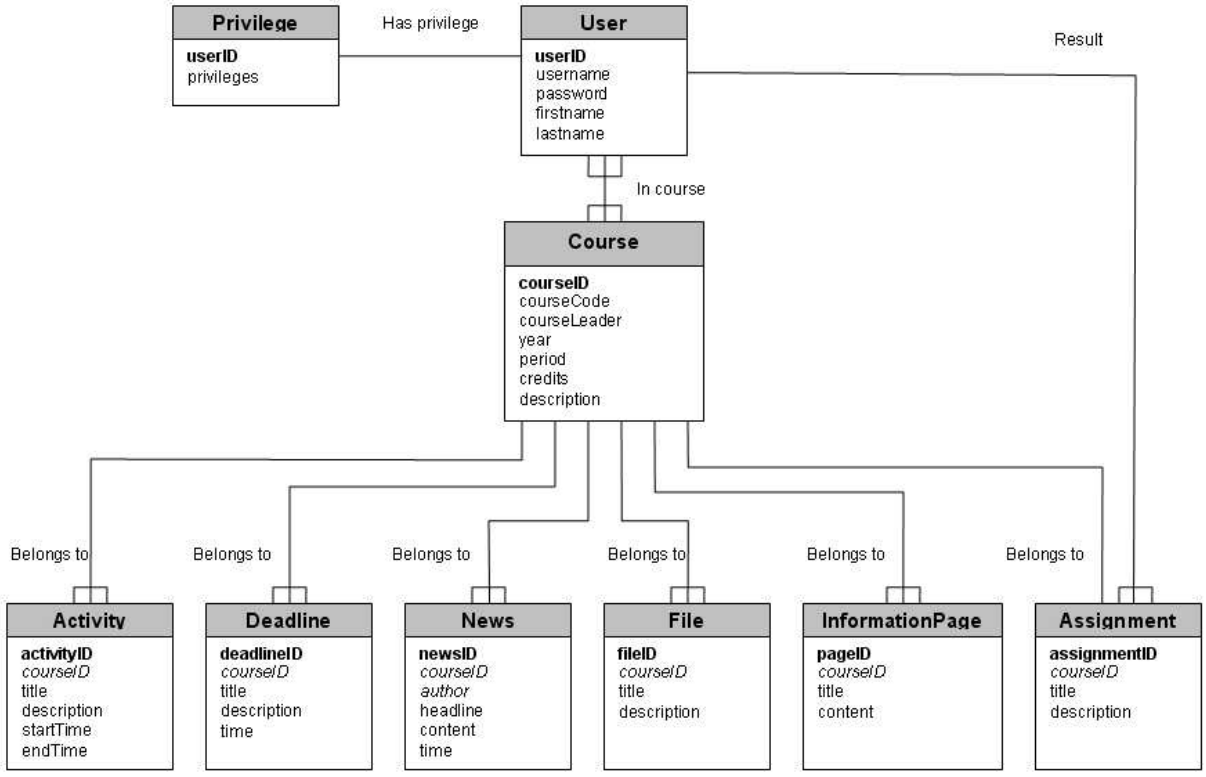
The controllers manipulate the data and contain the business logic of the system. To communicate with the database server the controllers use the Database Access Layer, which handles the connection and communication with the database.

The session is a controller that is used to identify different users. It is used throughout the system and is therefore illustrated in the diagram.

2.3 Detailed Architecture

2.3.1 Database System

Entity-relationship model



The above is an entity-relationship model (ER model) of the database structure that will be used by the system. Primary keys are distinguished by bold font, and foreign keys are distinguished by italic font.

The central part of data in the system is the *Course* table where information about the courses is stored. Much of the rest of data in the system is related to the *Course* table. A course can

for example have activities, deadlines, news, files, information pages and assignments. All these must be related to exactly one course.

Users can be in none, one or several courses, and a course can have none, one or several users in it. Users are in a course when they're applying to get registered for it, fully registered or are involved with teaching in the course. The user's status in a specific course is available in as a status field in the *InCourse* relation.

Privileges which are not course specific (system administrator privileges) are stored in the *Privilege* table.

Results are stored in the *Result* table and are related to one user and one assignment. Each assignment is in turn related to one course.

Files are stored in the file system, and only their metadata is stored in the database. No filename is stored in the database since files will be renamed so that their file-ID can be used to locate the file.

A schedule consists of several activities belonging to a course. A specific user's schedule can be found by finding the courses the student is in and then fetching the activities for that course.

T-matrix

Type	Name	I-Term(s)	E-Term(s)
Object	User	userID	username, password, firstname, lastname
Object	Course	courseID	courseCode, courseLeader, year, period, credits, description
Object	Activity	activityID	title, description, startTime, endTime
Object	Deadline	deadlineID	title, description, time
Object	News	newsID	author, headline, content, time
Object	File	fileID	title, description
Object	InformationPage	pageID	title, content
Object	Assignment	assignmentID	title, description
Object	Privilege	userID	privileges
1:N	ActivityBelongsTo	courseID, activityID	
1:N	DeadlineBelongsTo	courseID, deadlineID	
1:N	NewsBelongsTo	courseID, newsID	
1:N	FileBelongsTo	courseID, fileID	
1:N	nformationPageBelongsTo	courseID, pageID	
1:N	AssignmentBelongsTo	courseID, assignmentID	
N:N	InCourse	userID, courseID	status
N:N	Result	userID, assignmentID	grade

Above is the T-matrix for the database illustrated in the ER model.

Database Structure

Table	Attribute
User	((userID), username, password, firstname, lastname)
Course	((courseID), courseCode, courseLeader, year, period, credits, description)
Activity	((activityID), courseID, title, description, startTime, endTime)
Deadline	((deadlineID), courseID, title, description, time)
News	((newsID), courseID, author, headline, content, time)
File	((fileID), courseID, title, description)
InformationPage	((pageID), courseID, title, content)
Assignment	((assignmentID), courseID, title, description)
Privilege	((userID), privilege)
InCourse	((userID, courseID), status)
Result	((userID, assignmentID), grade)

This is the database structure after normalizing it from the T-matrix. Privileges which are not course specific are stored in a separate table to avoid wasting space since very few users (system administrators) will have any special privileges.

2.3.2 Sequence Diagram

The sequence diagram models the flow of logic within the system where a horizontal arrow represents the interaction between two objects. The dotted vertical lines represents the time, where the time flows from top to bottom.

We decided to create one sequence diagram, “Create Database Post”, for the use cases add course description, add deadline, add course leader, add privileges, etc. because they have similar sequence of actions. The same goes for the sequence diagrams “Edit Database Post”, “Delete Database Post” and the view sequence diagrams. There are two view sequence diagrams, one that describes the flow if a post can be retrieved from the cache and the other if the post can’t be retrieved from the cache and has to get it from the database server.

Log In

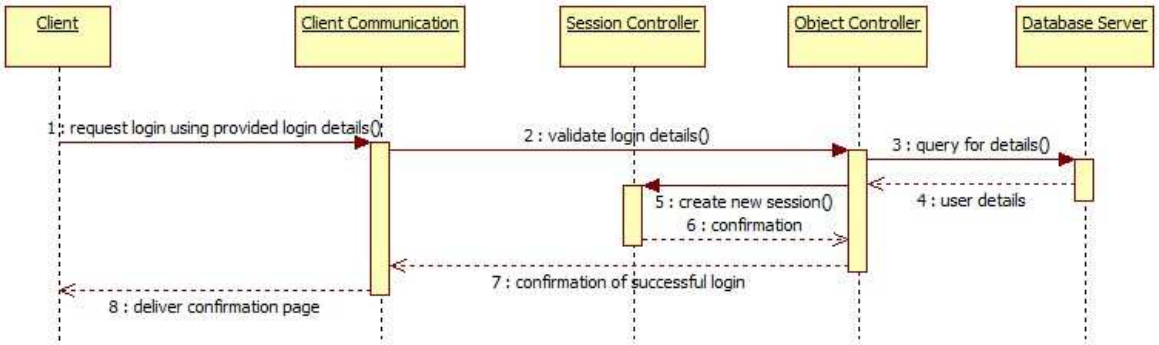


Figure 1 displays the sequence of action when a client requests to log in.

Create Database Post

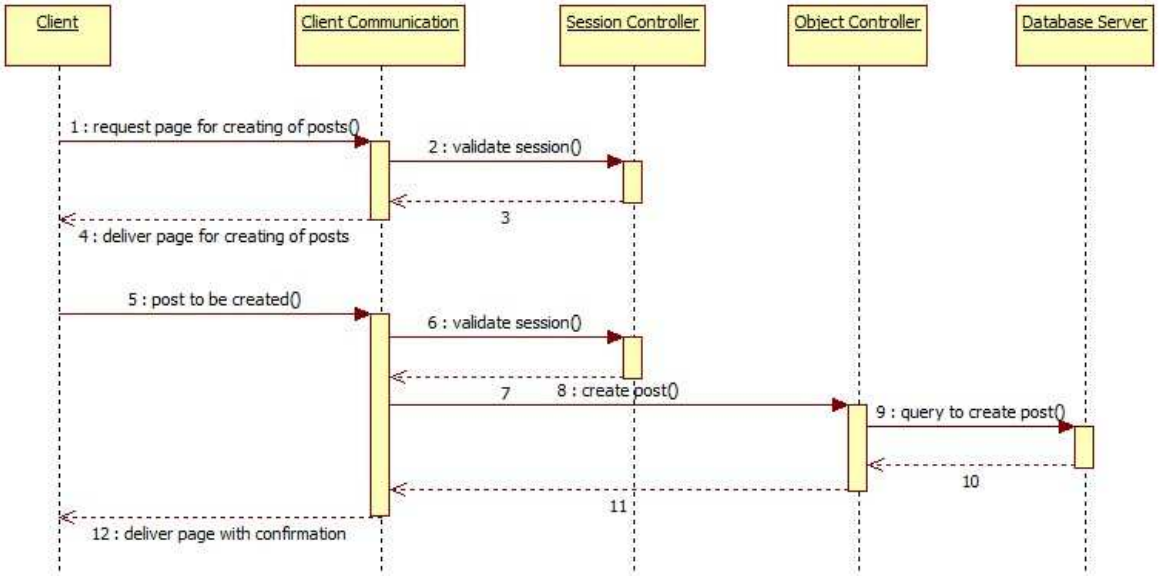


Figure 2 displays the sequence of actions to create a database post.

Edit Database Post

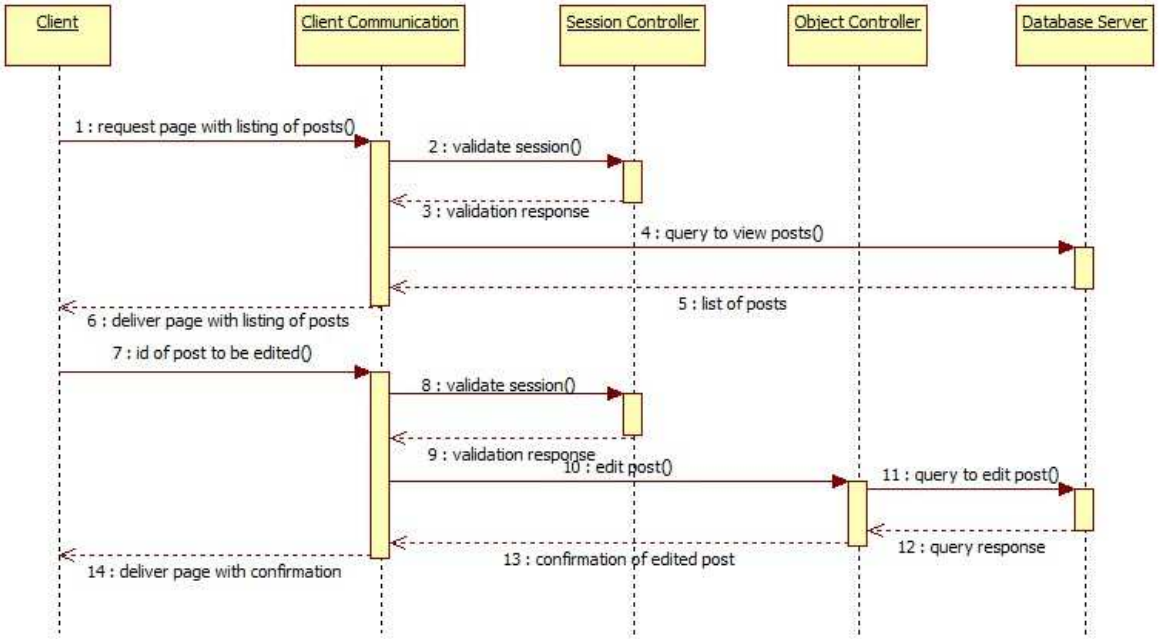


Figure 3 displays the sequence of actions to edit a database post.

Delete Database Post

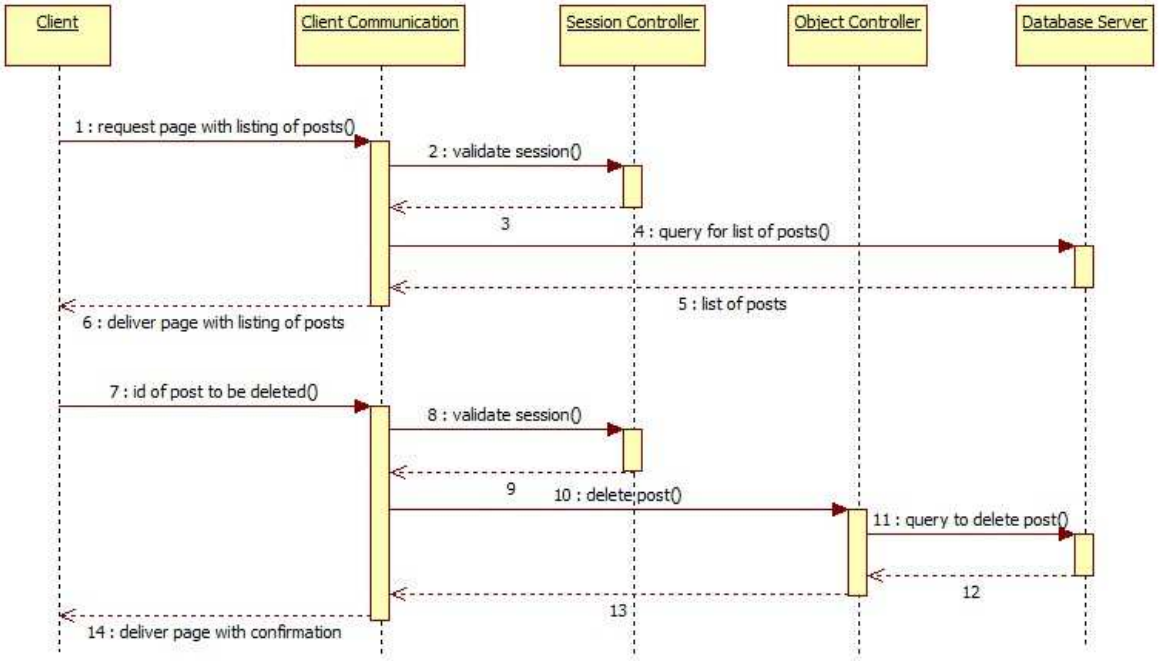


Figure 4 displays the sequence of actions to create a database post.

View Post from Cache

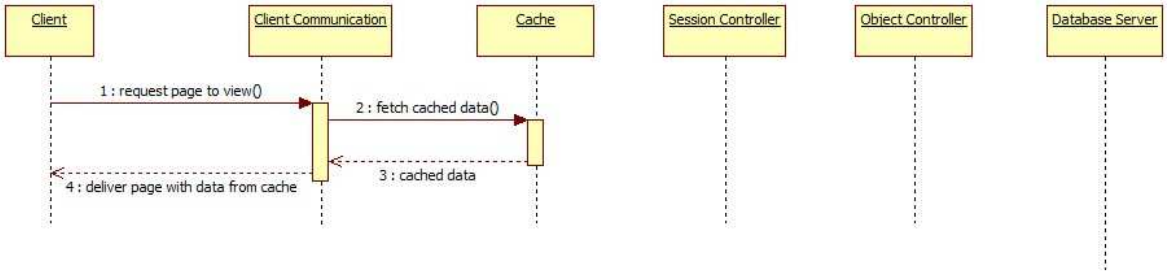


Figure 5 displays the sequence of actions to view a database post from cache.

View Post from Database

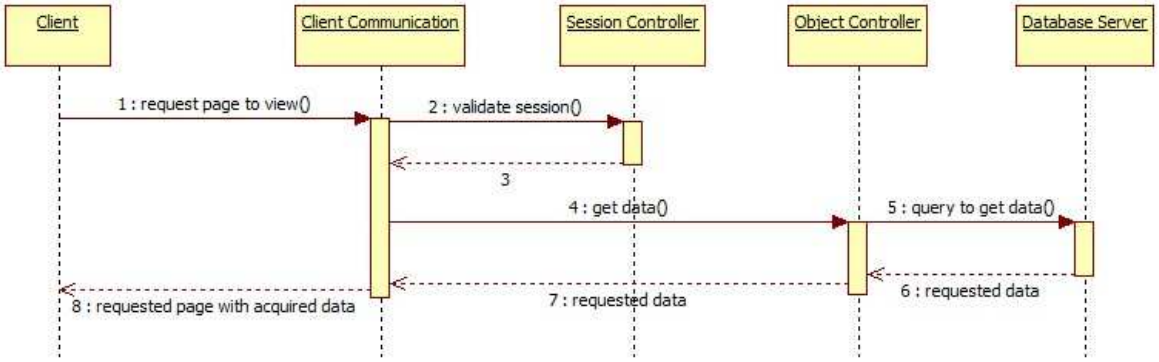


Figure 6 displays the sequence of actions to view a post from the database server when the post isn't available in the cache.

Export Schedule into iCalendar Format

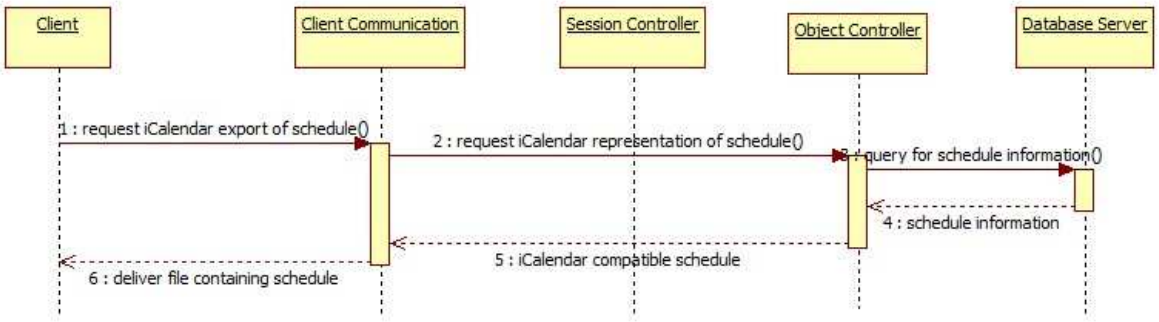


Figure 7 displays the sequence of actions to export a schedule into iCalander format.

Upload File

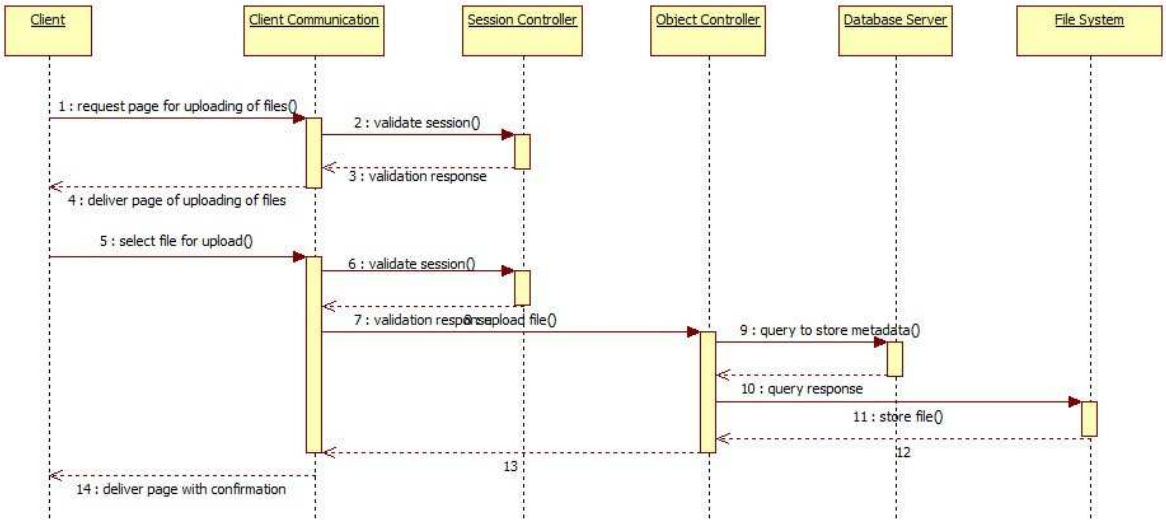


Figure 8 displays the sequence of actions when a client requests to upload a file.

Edit Uploaded File

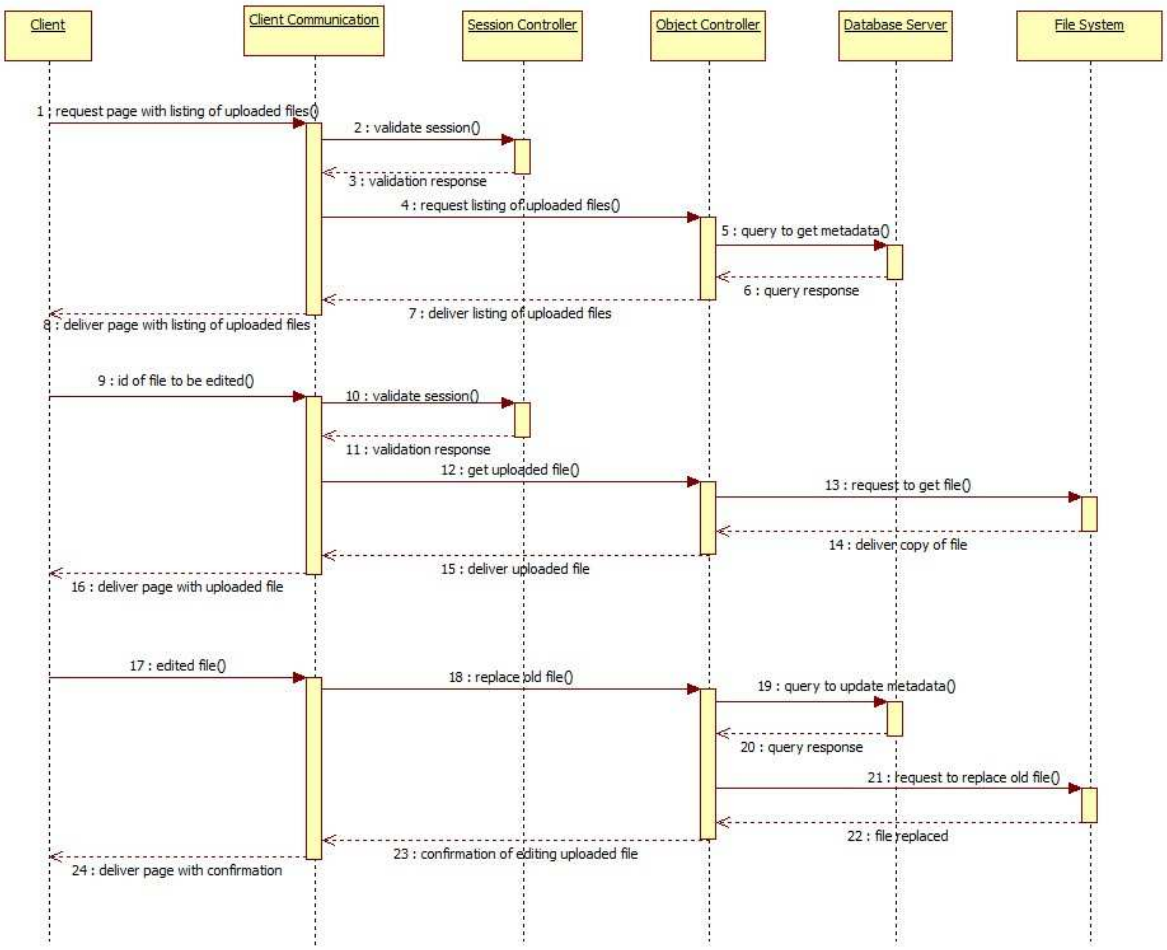


Figure 9 displays the sequence of actions when a client requests to edit an uploaded file.

Delete Uploaded File

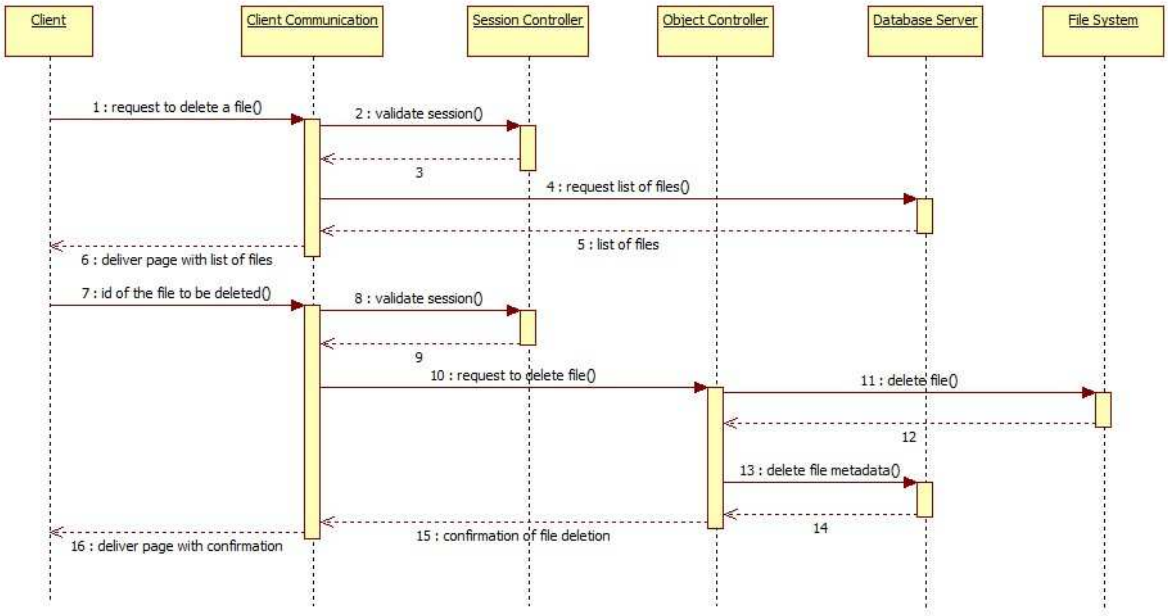


Figure 10 displays the sequence of actions when a client requests to delete an uploaded file