Converge

SOFTWARE ENGINEERING(IT-314)

Prof. Saurabh Tiwari

Group - 23

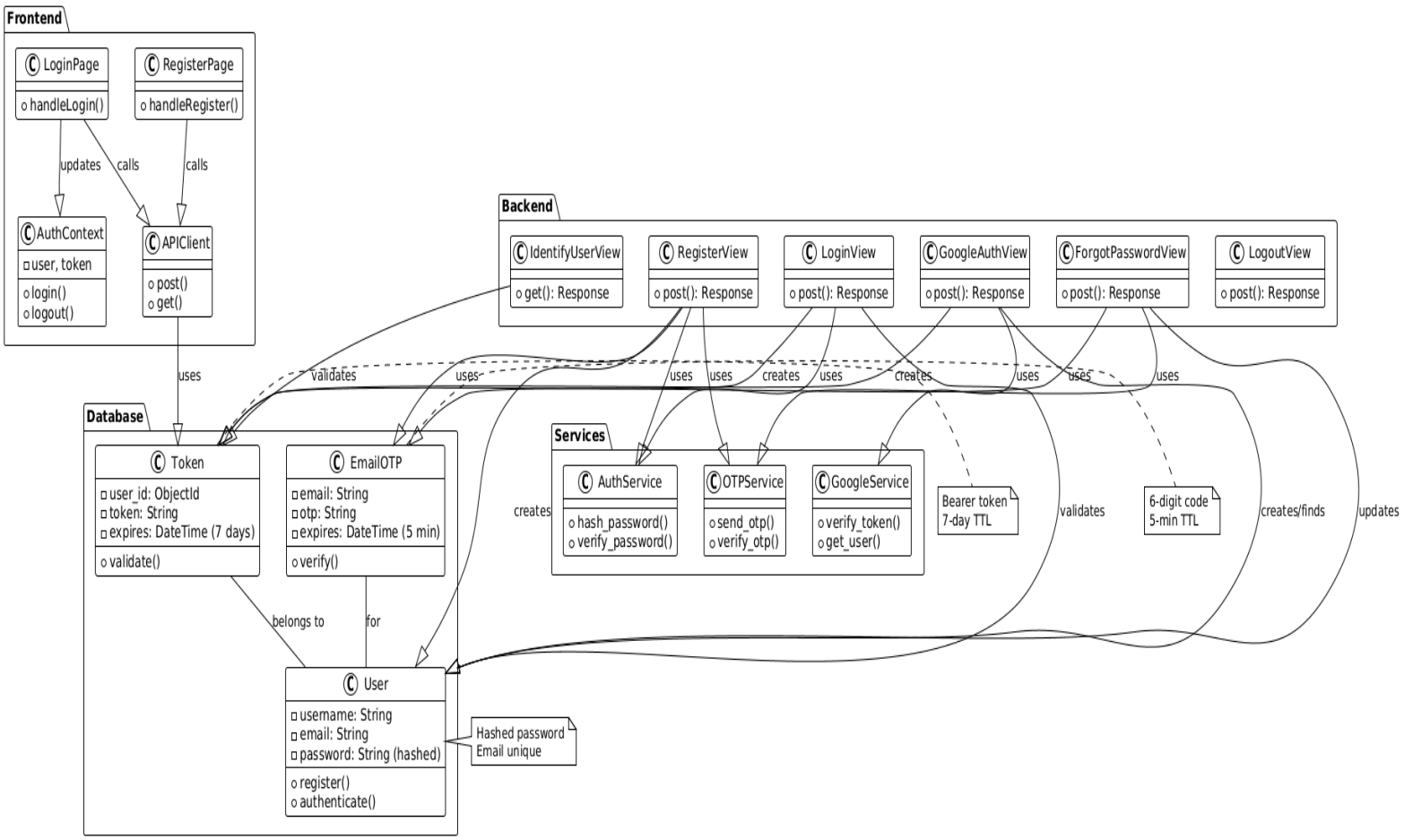
- 202301209- Gori Faran Firozbhai (Leader)
- 202301224- Prajapati Anushka Maheshbhai
- 202301244- Atharv Shah
- 202301213- Kalriya Prayag Jagdishbhai
- 202301251- Vasani Devarsh Chintanbhai
- 202301243- Vadaviya Chaitri Hareshkumar
- 202301212- Patel Ishti Upendra
- 202301211- Rathod Harita Rajeshbhai
- 202301228- Patel Naitikkumar Dilipbhai
- 202301242- Tejani Charvik Bipinbhai

Sprints:

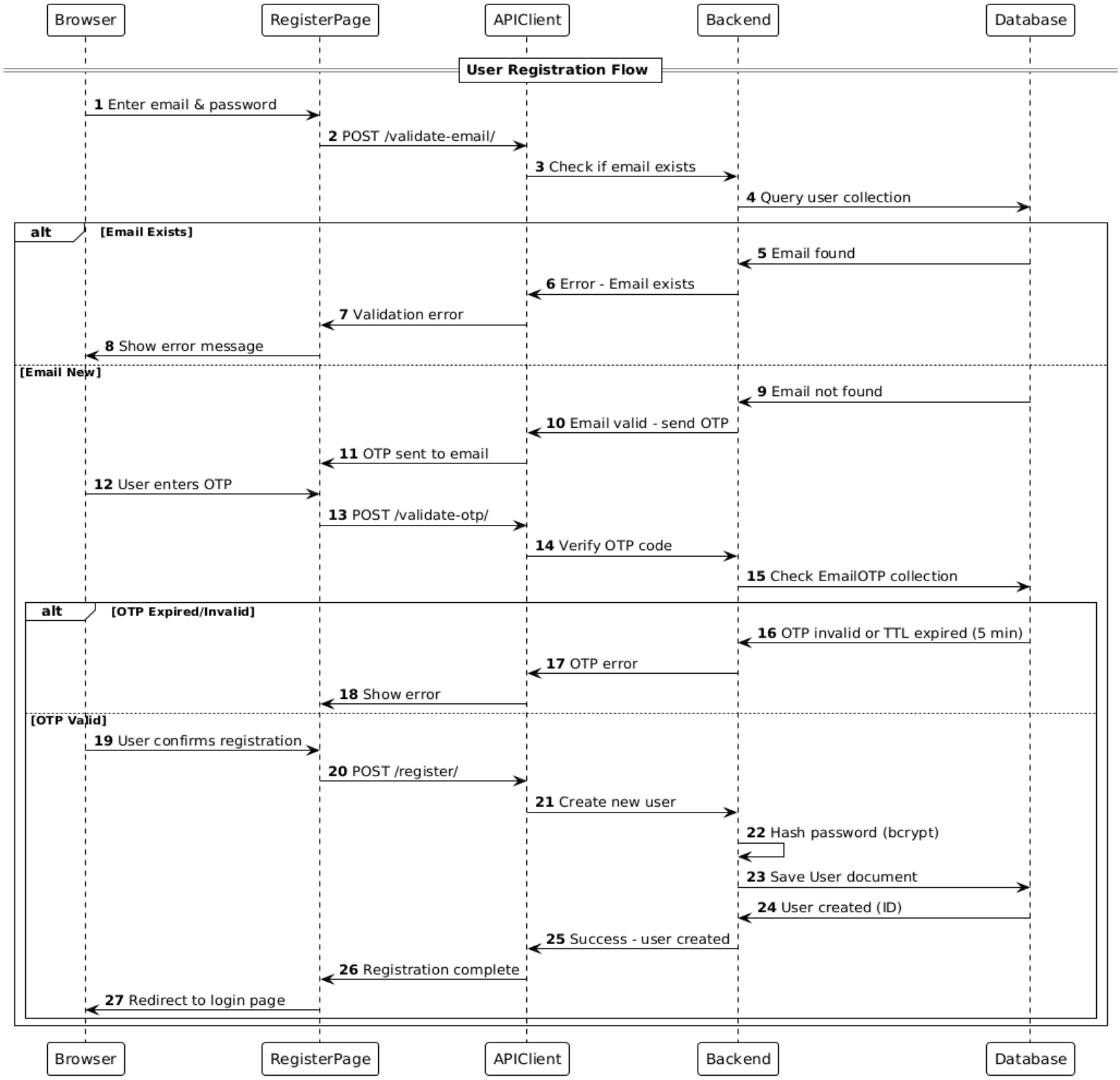
Sprint 1– Authentication

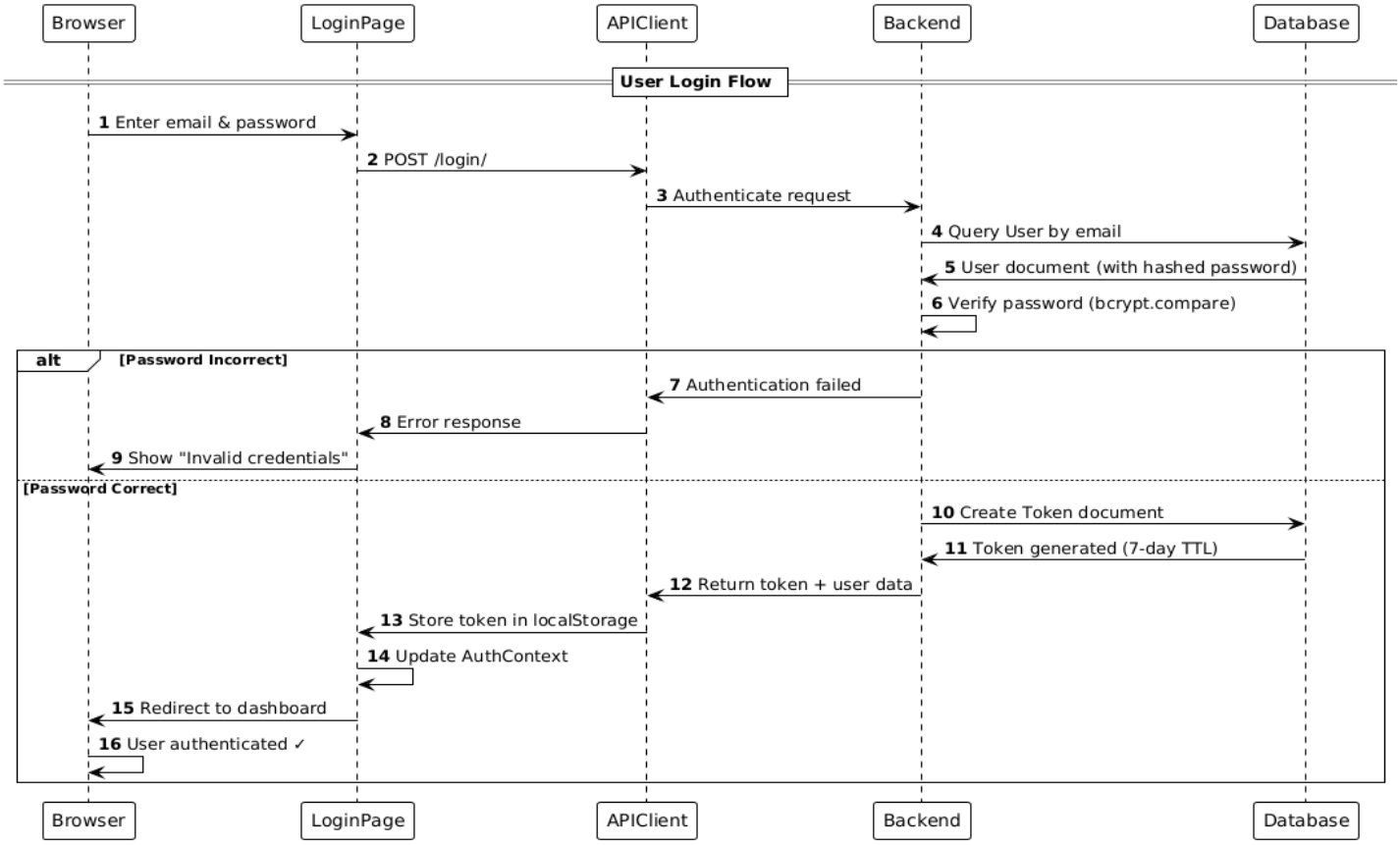
- **Sprint Goal:** Establish secure access for users.
- **Duration:** 1 week.
- **Deliverables:**
  - Secure authentication module.
  - User Register/login/logout/Google login functionality.
- **Key User Story:** US-21

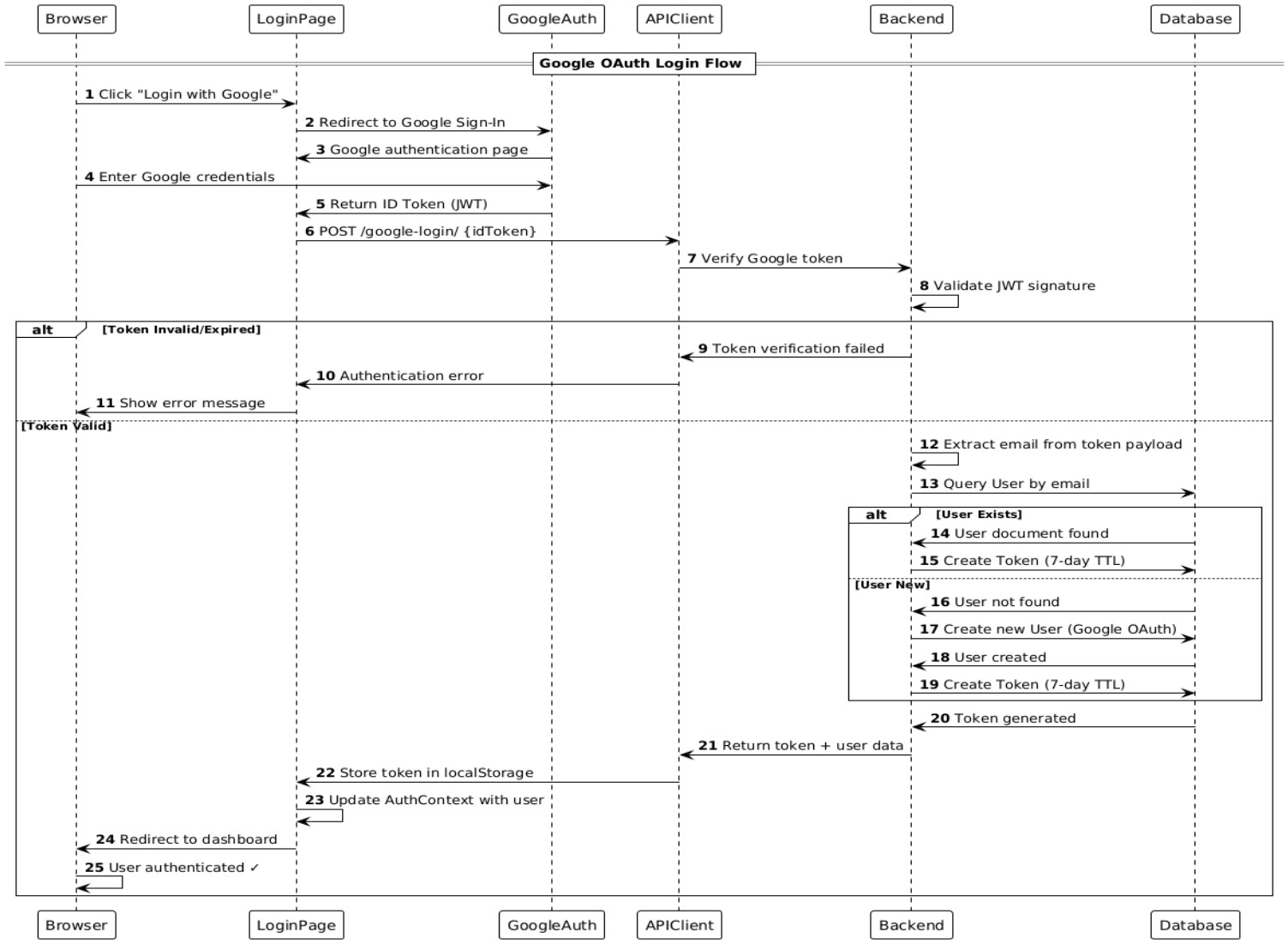
- **Class Diagram:**

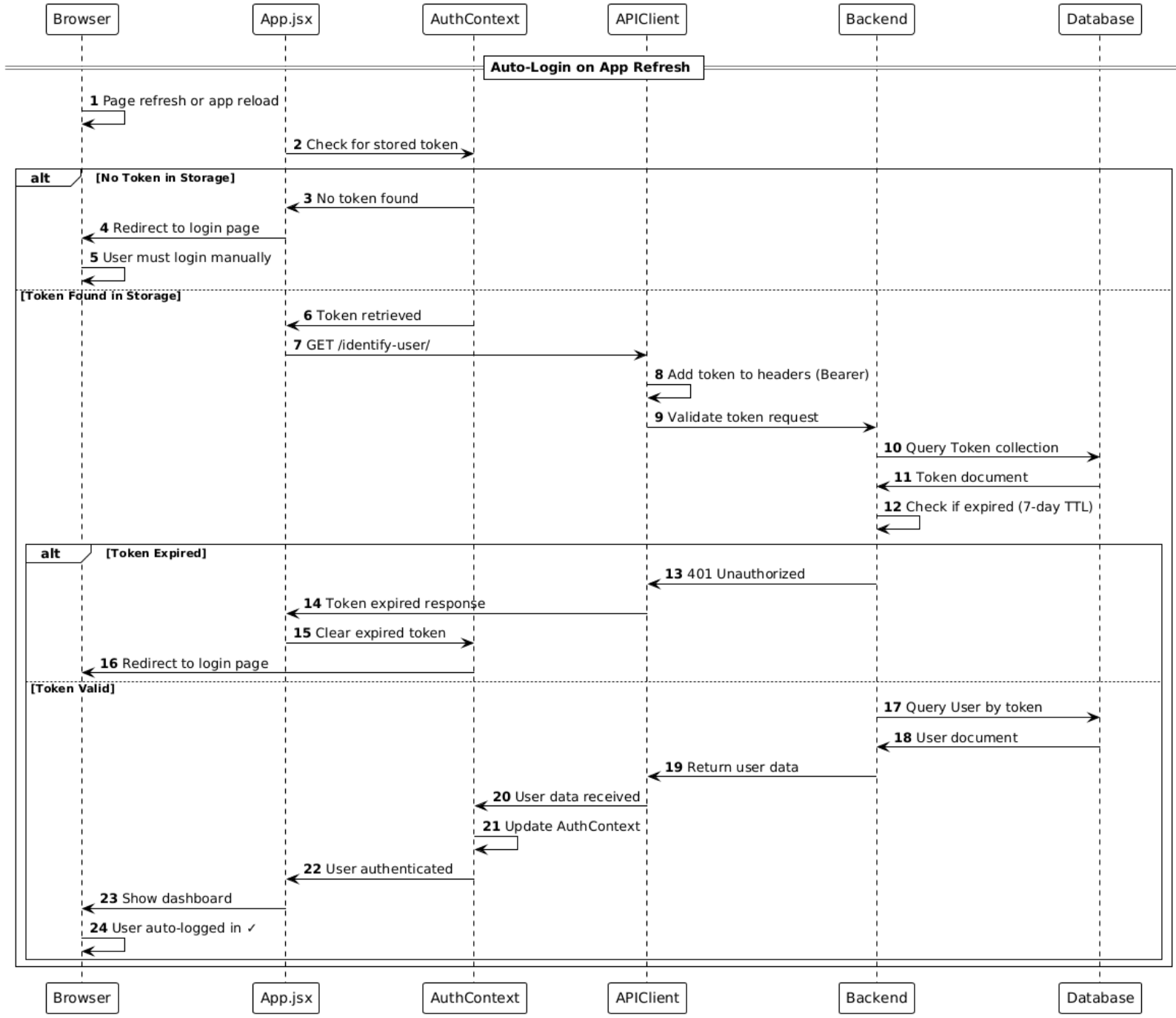


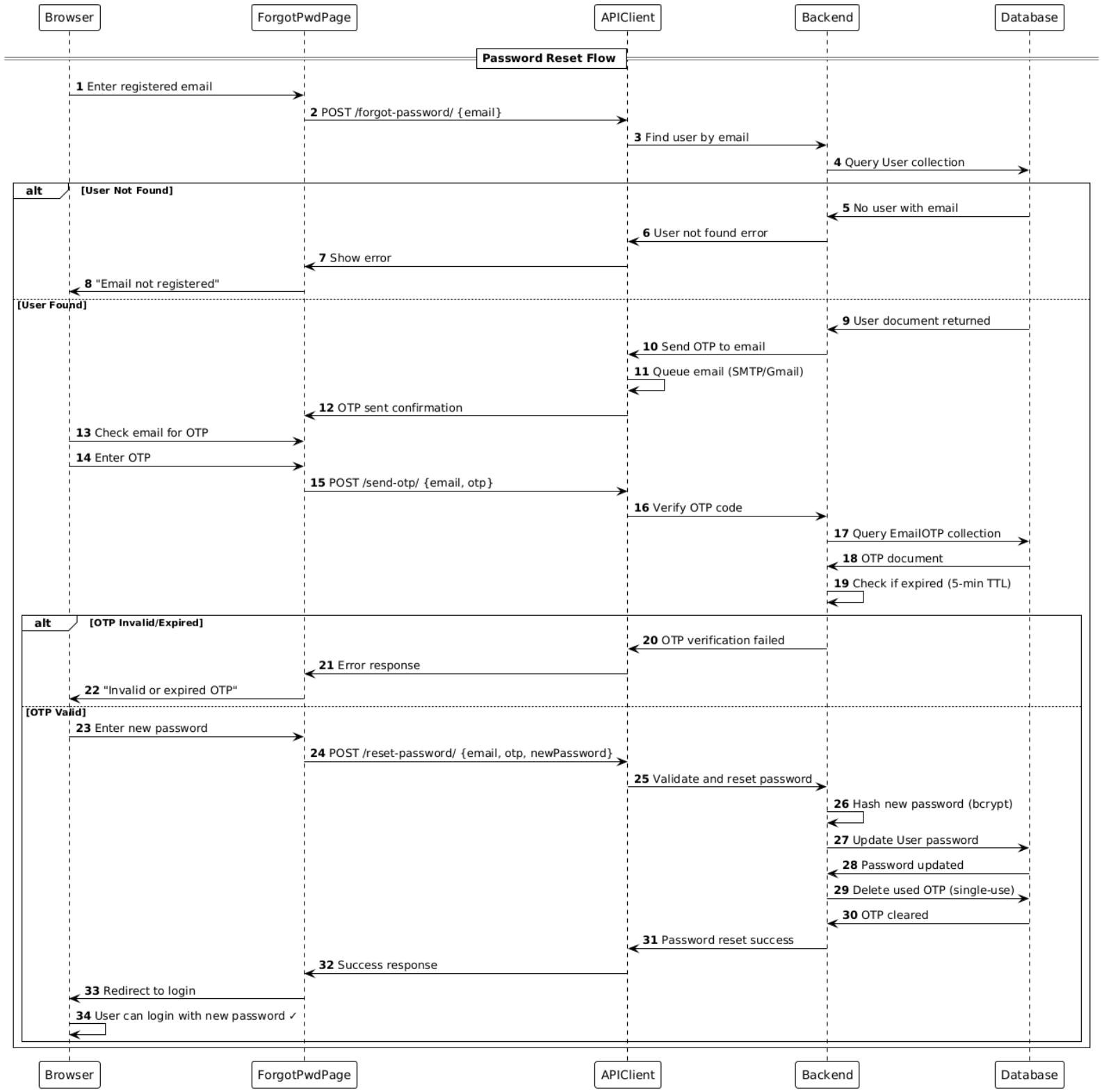
- **Sequence Diagrams:**




Browser
LoginPage
APIClient
Backend
Database
User Login Flow
1 Enter email & password
2 POST /login/
3 Authenticate request
4 Query User by email
5 User document (with hashed password)
6 Verify password (bcrypt.compare)
alt
[Password Incorrect]
7 Authentication failed
8 Error response
9 Show "Invalid credentials"
[Password Correct]
10 Create Token document
11 Token generated (7-day TTL)
12 Return token + user data
13 Store token in localStorage
14 Update AuthContext
15 Redirect to dashboard
16 User authenticated ✓
Browser
LoginPage
APIClient
Backend
Database


Browser
LoginPage
GoogleAuth
APIClient
Backend
Database
Google OAuth Login Flow
1 Click "Login with Google"
2 Redirect to Google Sign-In
3 Google authentication page
4 Enter Google credentials
5 Return ID Token (JWT)
6 POST /google-login/ {idToken}
7 Verify Google token
8 Validate JWT signature
alt
[Token Invalid/Expired]
9 Token verification failed
10 Authentication error
11 Show error message
[Token Valid]
12 Extract email from token payload
13 Query User by email
alt
[User Exists]
14 User document found
15 Create Token (7-day TTL)
[User New]
16 User not found
17 Create new User (Google OAuth)
18 User created
19 Create Token (7-day TTL)
20 Token generated
21 Return token + user data
22 Store token in localStorage
23 Update AuthContext with user
24 Redirect to dashboard
25 User authenticated ✓
Browser
LoginPage
GoogleAuth
APIClient
Backend
Database

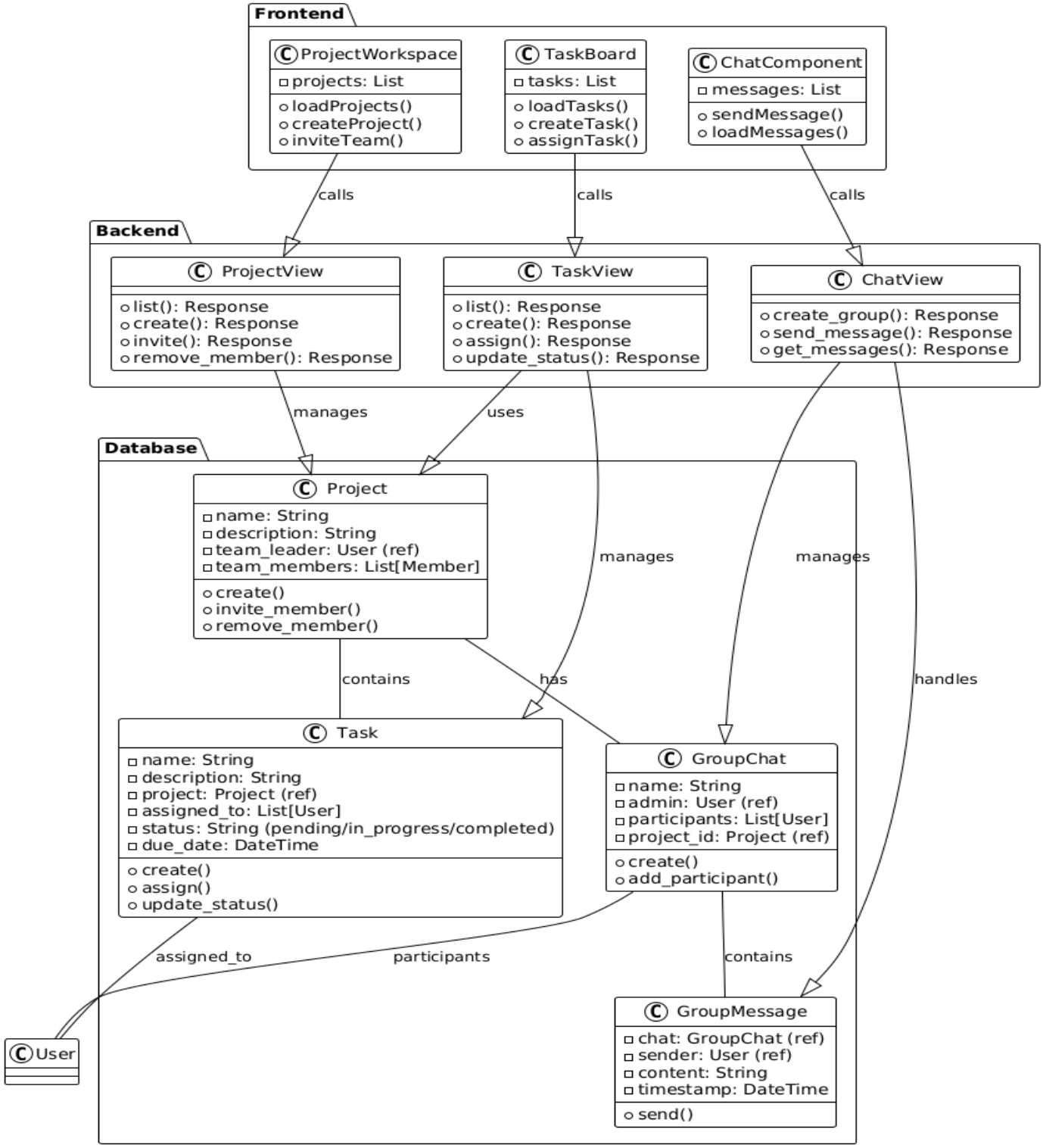

Browser
App.jsx
AuthContext
APIClient
Backend
Database
Auto-Login on App Refresh
1 Page refresh or app reload
2 Check for stored token
alt
[No Token in Storage]
3 No token found
4 Redirect to login page
5 User must login manually
[Token Found in Storage]
6 Token retrieved
7 GET /identify-user/
8 Add token to headers (Bearer)
9 Validate token request
10 Query Token collection
11 Token document
12 Check if expired (7-day TTL)
alt
[Token Expired]
13 401 Unauthorized
14 Token expired response
15 Clear expired token
16 Redirect to login page
[Token Valid]
17 Query User by token
18 User document
19 Return user data
20 User data received
21 Update AuthContext
22 User authenticated
23 Show dashboard
24 User auto-logged in ✓
Browser
App.jsx
AuthContext
APIClient
Backend
Database


Browser
ForgotPwdPage
APIClient
Backend
Database
Password Reset Flow
1 Enter registered email
2 POST /forgot-password/ {email}
3 Find user by email
4 Query User collection
alt
[User Not Found]
5 No user with email
6 User not found error
7 Show error
8 "Email not registered"
[User Found]
9 User document returned
10 Send OTP to email
11 Queue email (SMTP/Gmail)
12 OTP sent confirmation
13 Check email for OTP
14 Enter OTP
15 POST /send-otp/ {email, otp}
16 Verify OTP code
17 Query EmailOTP collection
18 OTP document
19 Check if expired (5-min TTL)
alt
[OTP Invalid/Expired]
20 OTP verification failed
21 Error response
22 "Invalid or expired OTP"
[OTP Valid]
23 Enter new password
24 POST /reset-password/ {email, otp, newPassword}
25 Validate and reset password
26 Hash new password (bcrypt)
27 Update User password
28 Password updated
29 Delete used OTP (single-use)
30 OTP cleared
31 Password reset success
32 Success response
33 Redirect to login
34 User can login with new password ✓
Browser
ForgotPwdPage
APIClient
Backend
Database

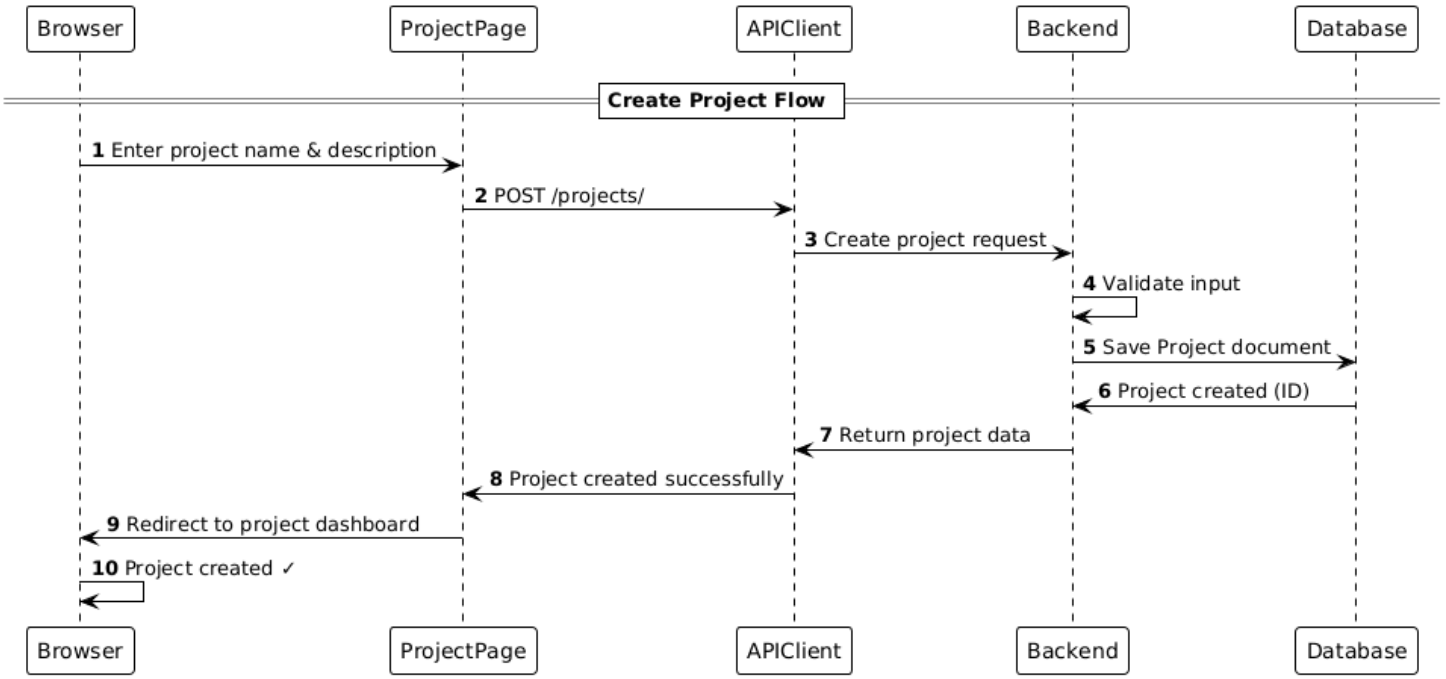
Sprint 2– Project Setup & Team Communication

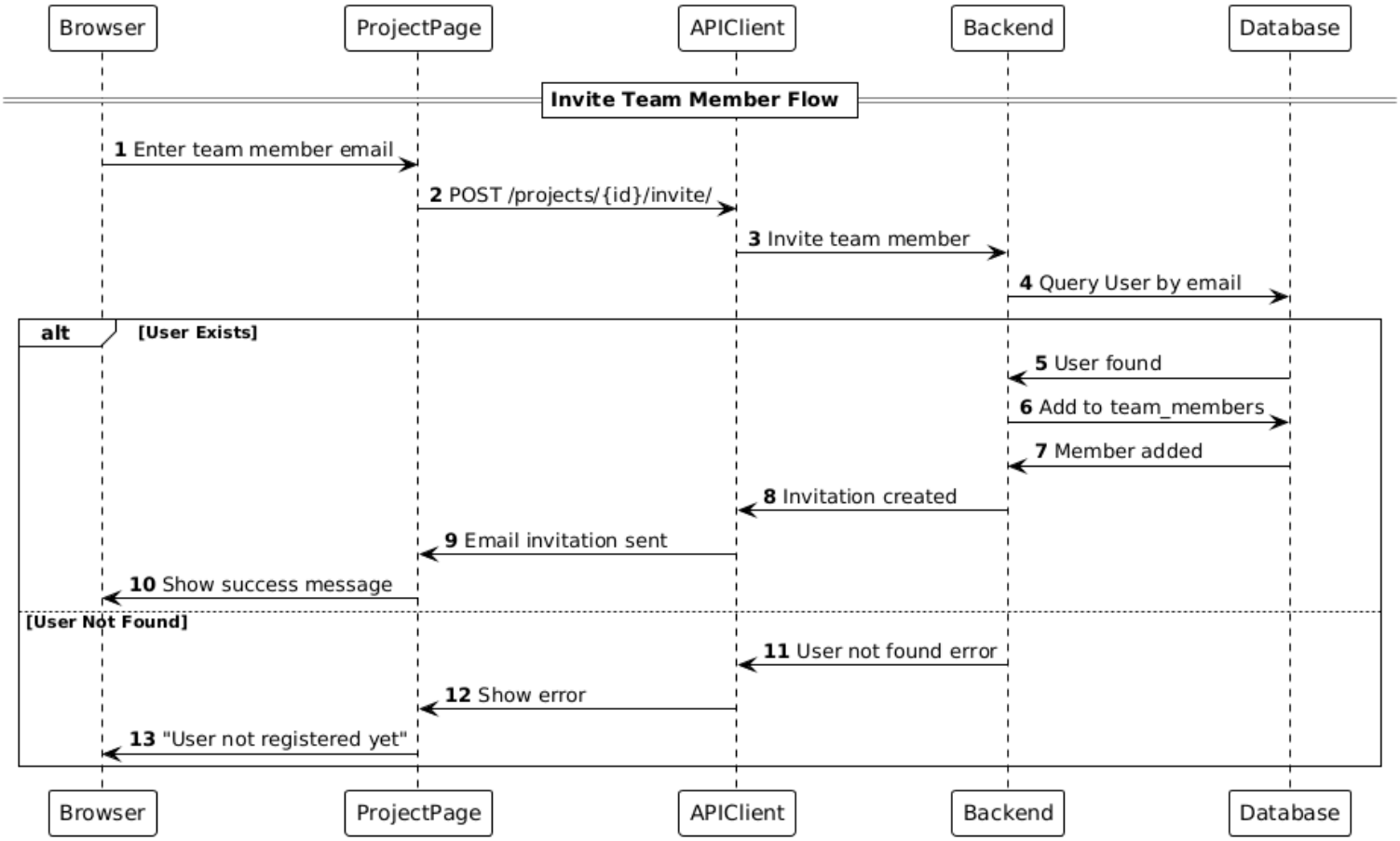
- **Sprint Goal:** Enable project creation, team onboarding, and basic collaboration.
- **Duration:** 1.5 week.
- **Deliverables:**
  - Create and manage projects.
  - Invite team members.
  - Task creation, assignment, and updates.
  - Group chat for team discussions.
- **Key User Story:** US-7

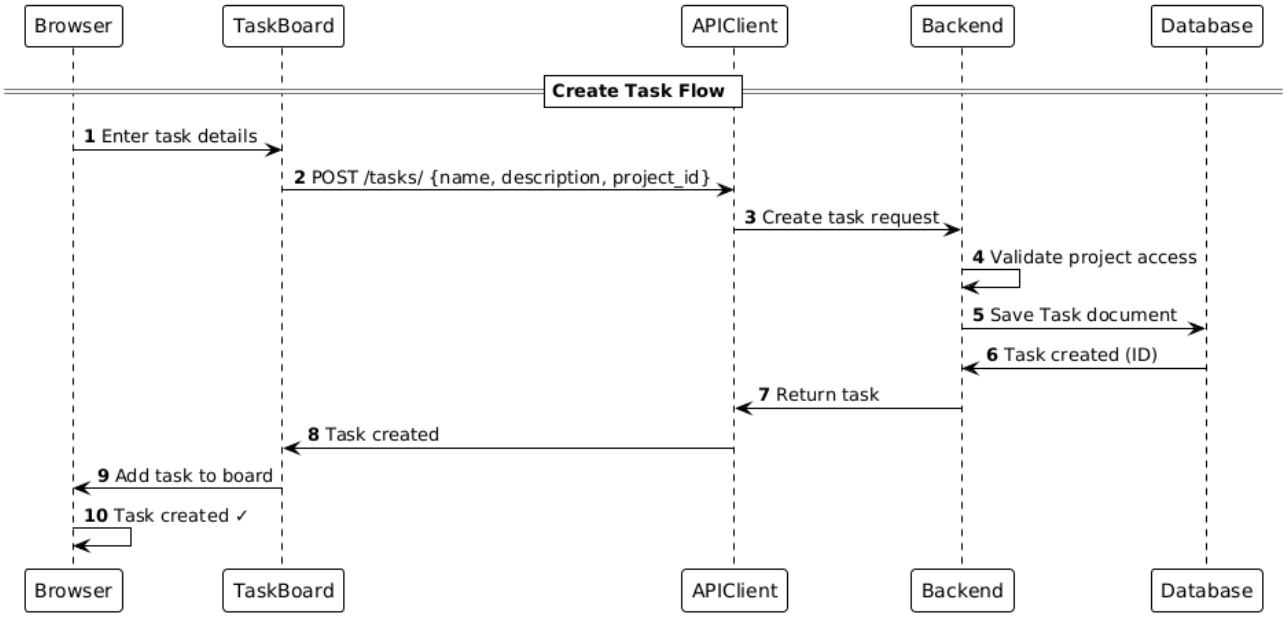
- ## Class Diagram:

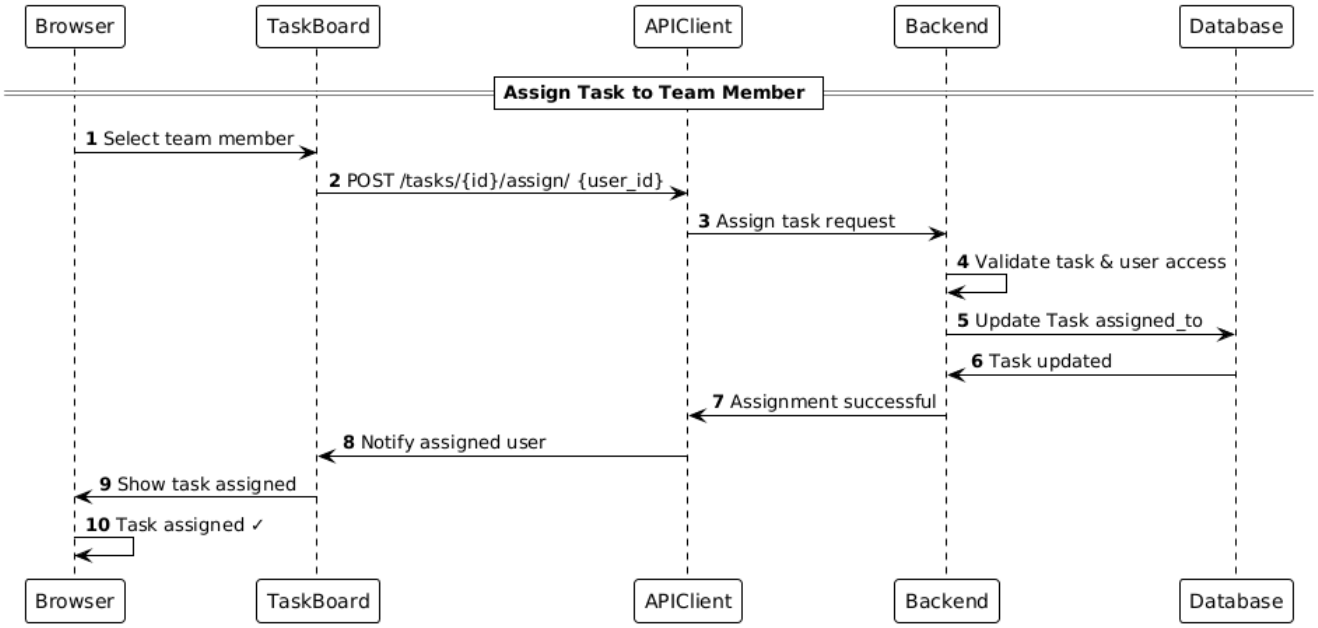


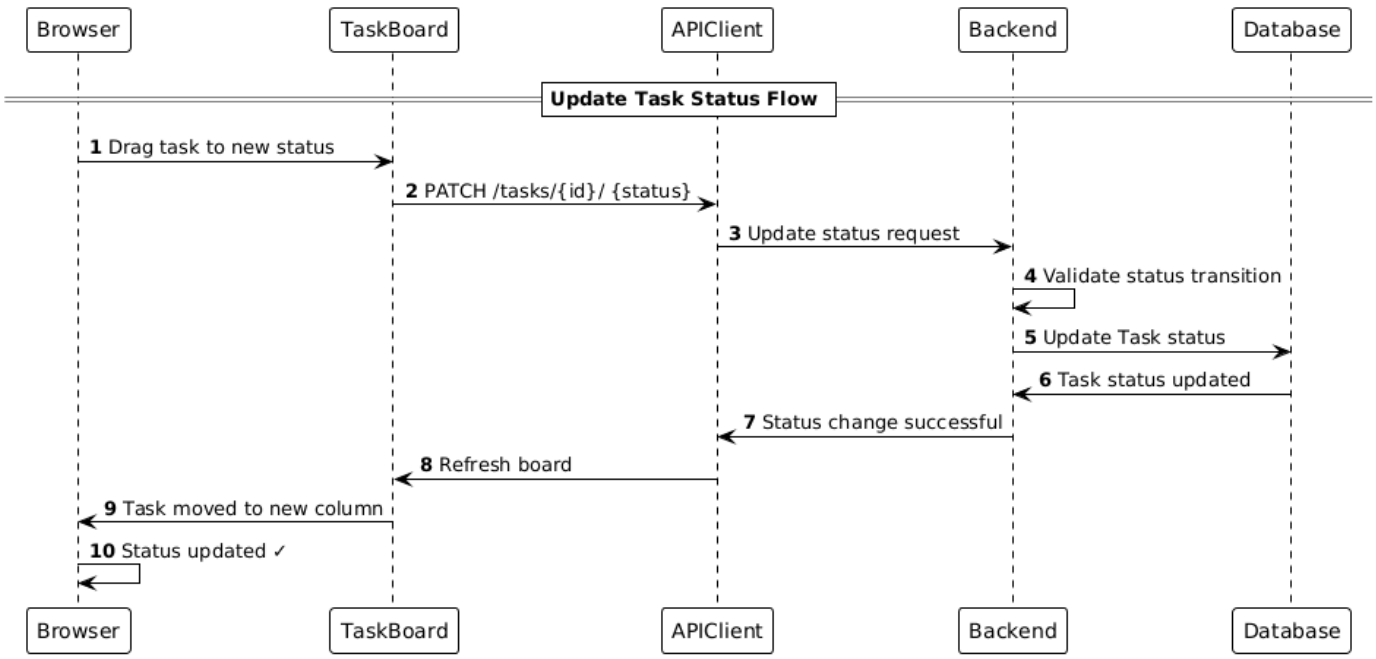
- **Sequence Diagrams:**

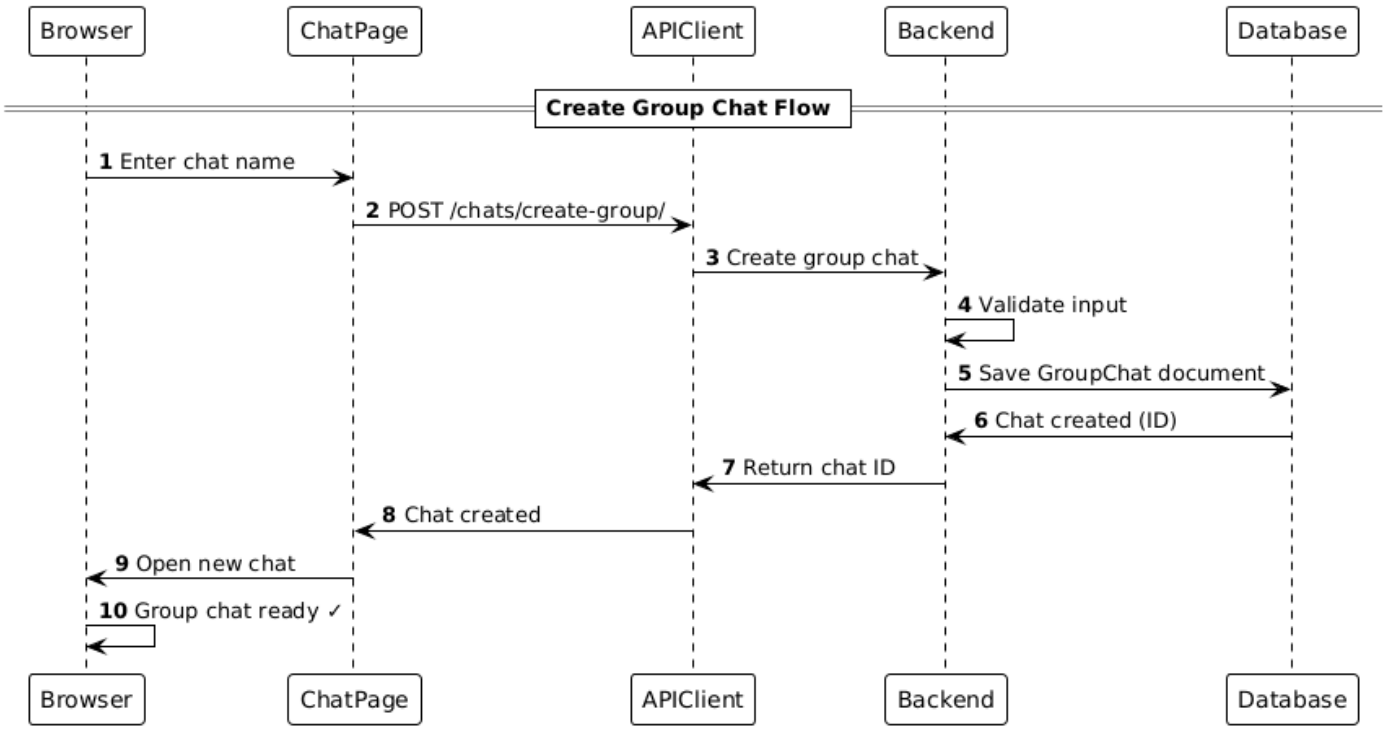


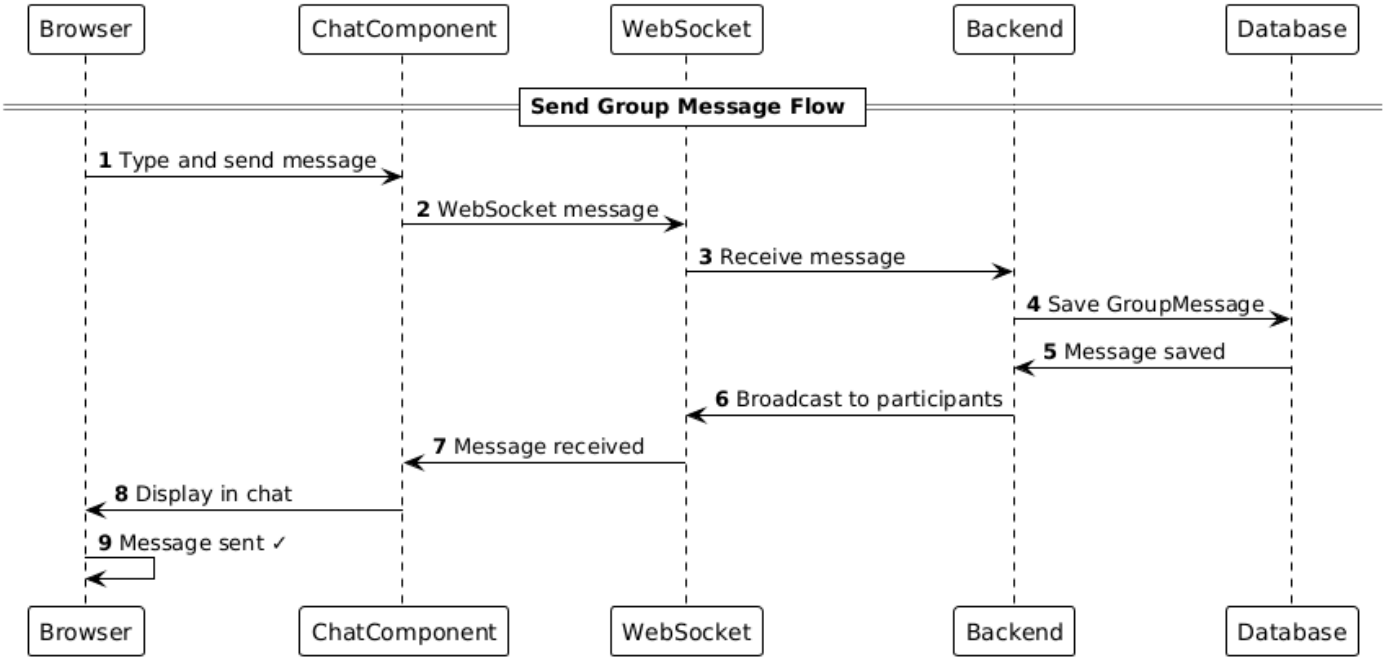



Browser
TaskBoard
APIClient
Backend
Database
Create Task Flow
1 Enter task details
2 POST /tasks/ {name, description, project_id}
3 Create task request
4 Validate project access
5 Save Task document
6 Task created (ID)
7 Return task
8 Task created
9 Add task to board
10 Task created ✓
Browser
TaskBoard
APIClient
Backend
Database


Browser
TaskBoard
APIClient
Backend
Database
Assign Task to Team Member
1 Select team member
2 POST /tasks/{id}/assign/ {user_id}
3 Assign task request
4 Validate task & user access
5 Update Task assigned_to
6 Task updated
7 Assignment successful
8 Notify assigned user
9 Show task assigned
10 Task assigned ✓
Browser
TaskBoard
APIClient
Backend
Database


Browser
TaskBoard
APIClient
Backend
Database
Update Task Status Flow
1 Drag task to new status
2 PATCH /tasks/{id}/ {status}
3 Update status request
4 Validate status transition
5 Update Task status
6 Task status updated
7 Status change successful
8 Refresh board
9 Task moved to new column
10 Status updated ✓
Browser
TaskBoard
APIClient
Backend
Database

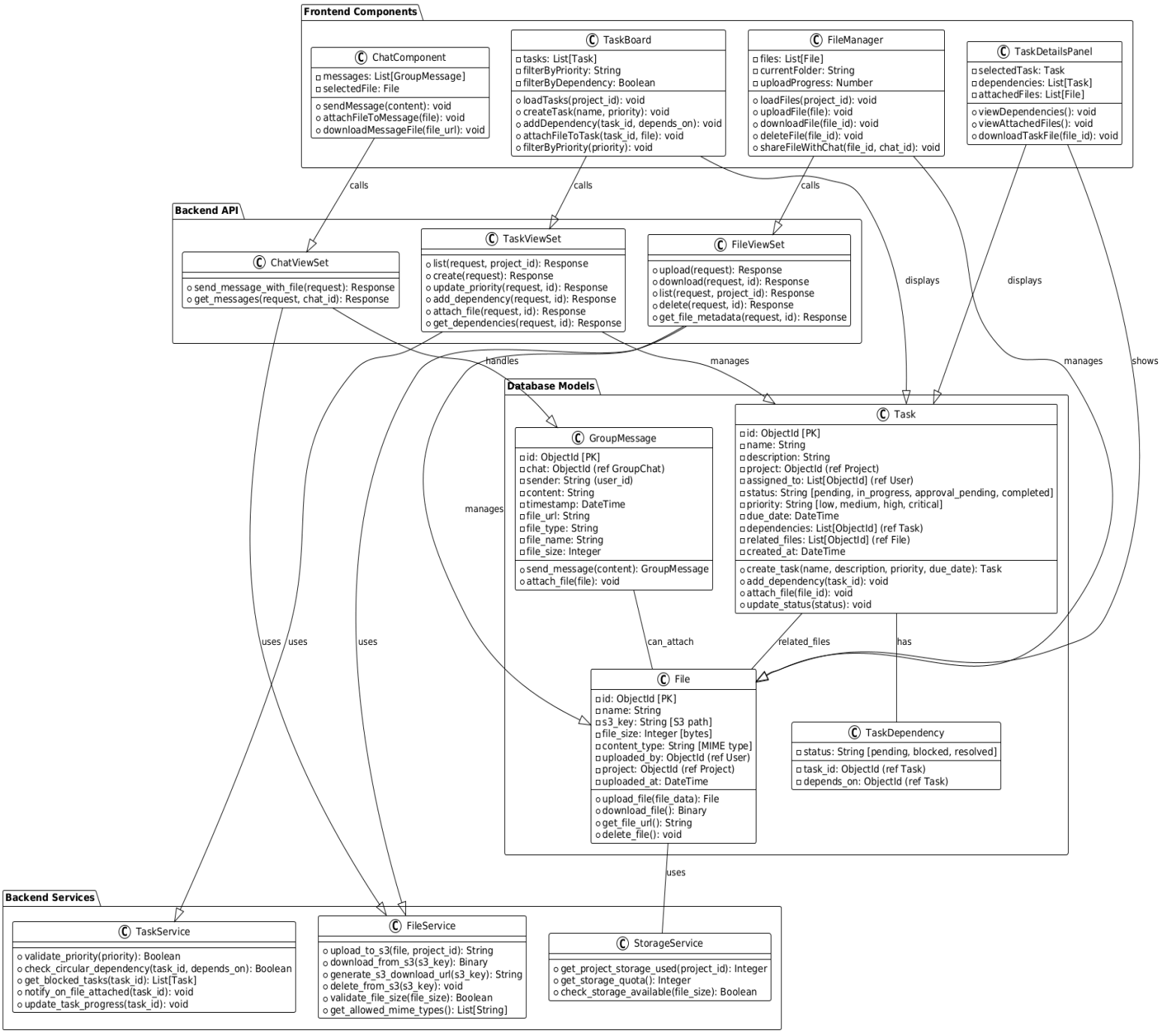

Browser
ChatPage
APIClient
Backend
Database
Create Group Chat Flow
1 Enter chat name
2 POST /chats/create-group/
3 Create group chat
4 Validate input
5 Save GroupChat document
6 Chat created (ID)
7 Return chat ID
8 Chat created
9 Open new chat
10 Group chat ready ✓
Browser
ChatPage
APIClient
Backend
Database


Browser
ChatComponent
WebSocket
Backend
Database
Send Group Message Flow
1 Type and send message
2 WebSocket message
3 Receive message
4 Save GroupMessage
5 Message saved
6 Broadcast to participants
7 Message received
8 Display in chat
9 Message sent ✓
Browser
ChatComponent
WebSocket
Backend
Database

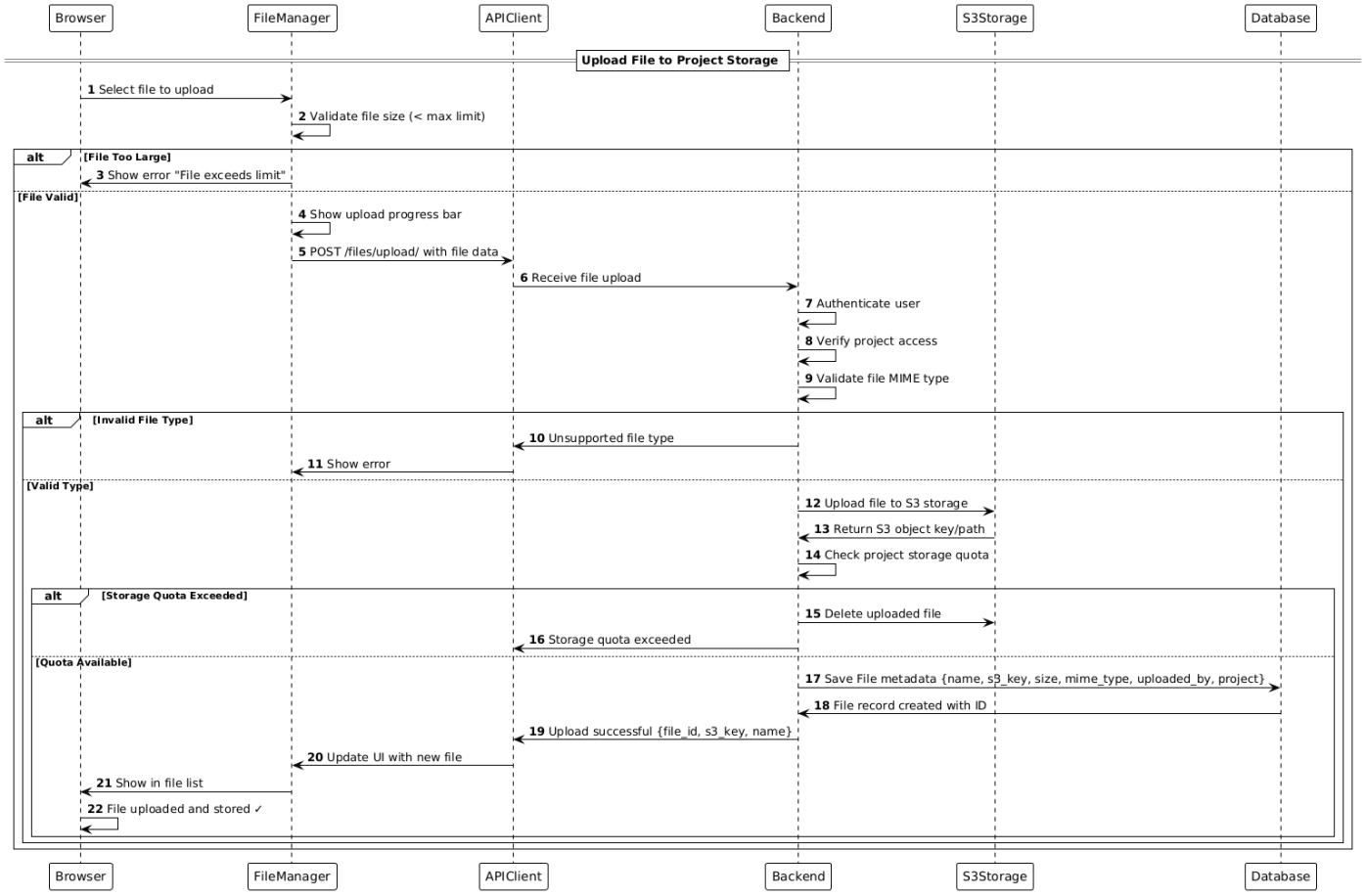
Sprint 3– Task & File Management

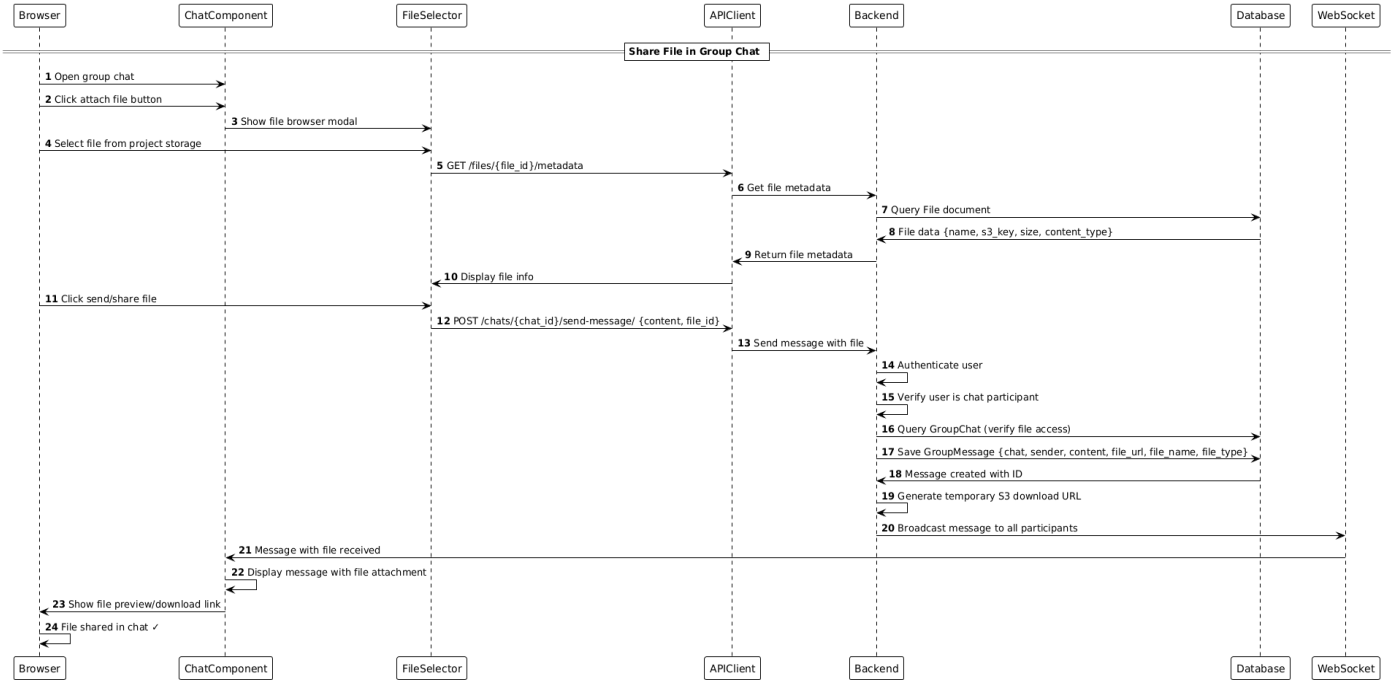
- **Sprint Goal:** Enhance productivity with task tracking and file management.
- **Duration:** 1.5 week.
- **Deliverables:**
  - File sharing within chat.
  - Centralized file storage.
  - Task tracking, priorities, dependencies.
- **Key User Story:** US-1, US-2, US-3, US-4, US-5, US-10, US-13, US-15, US-16, US-17.

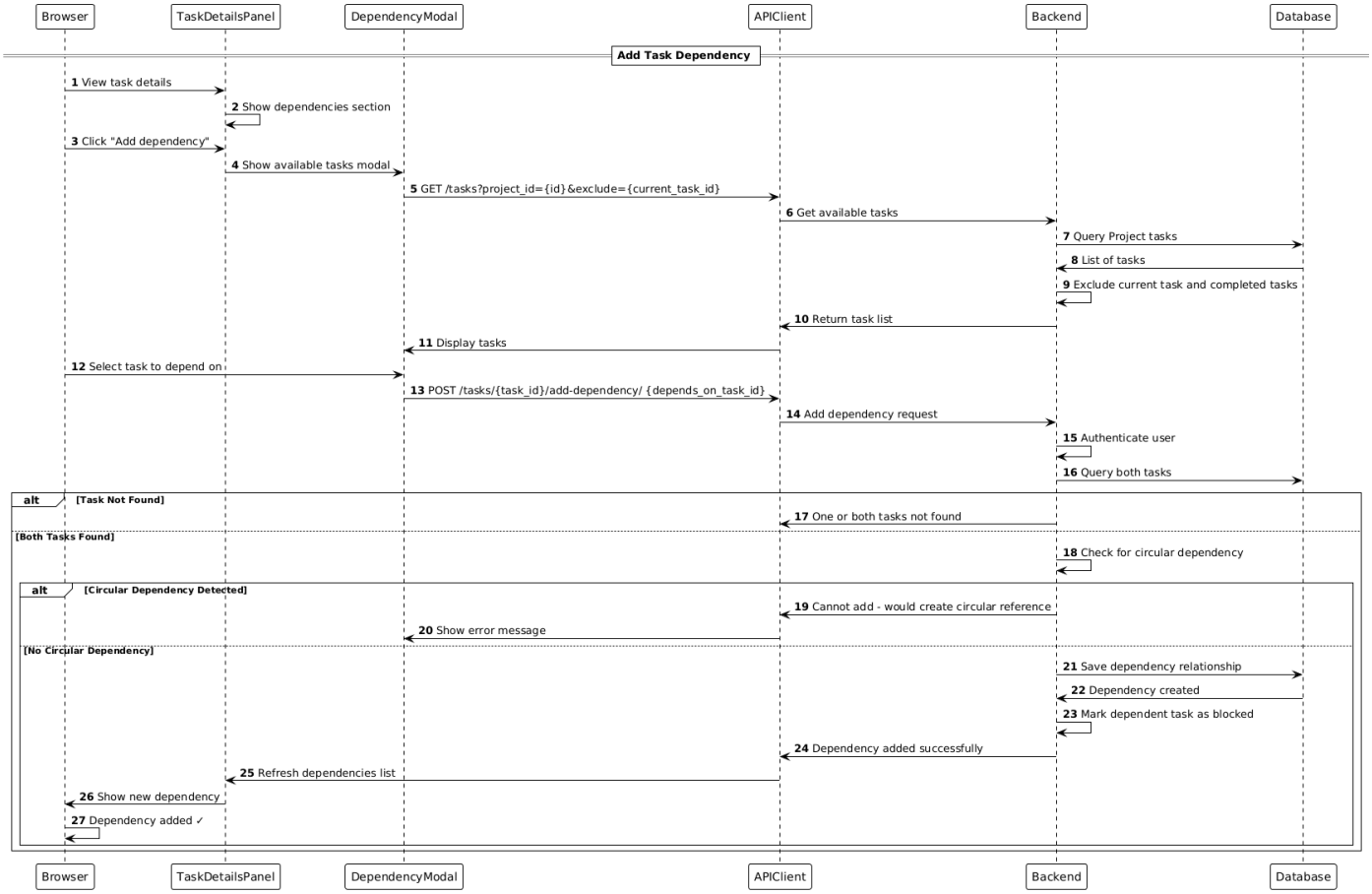
- ## Class Diagram:

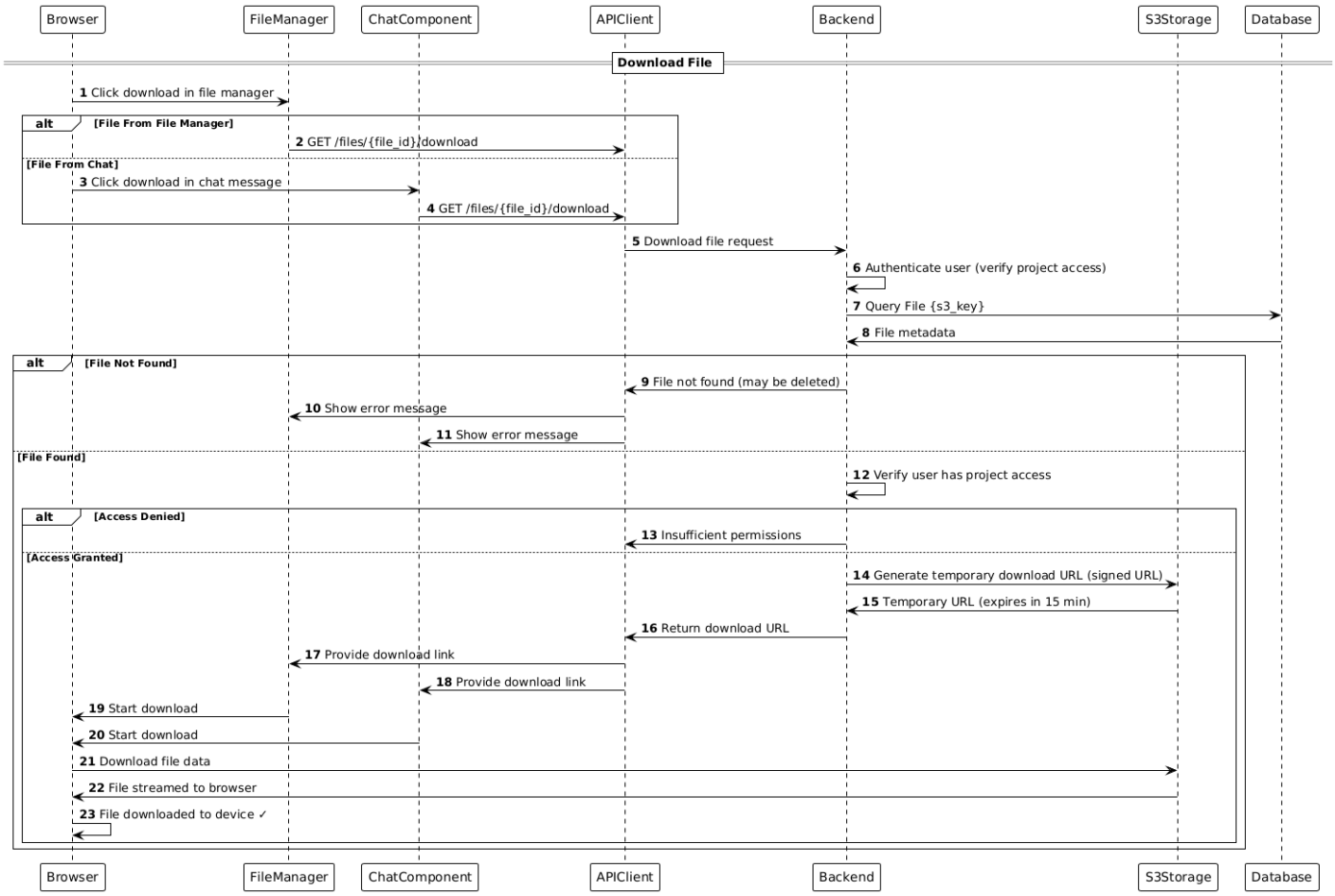


- **Sequence Diagrams:**





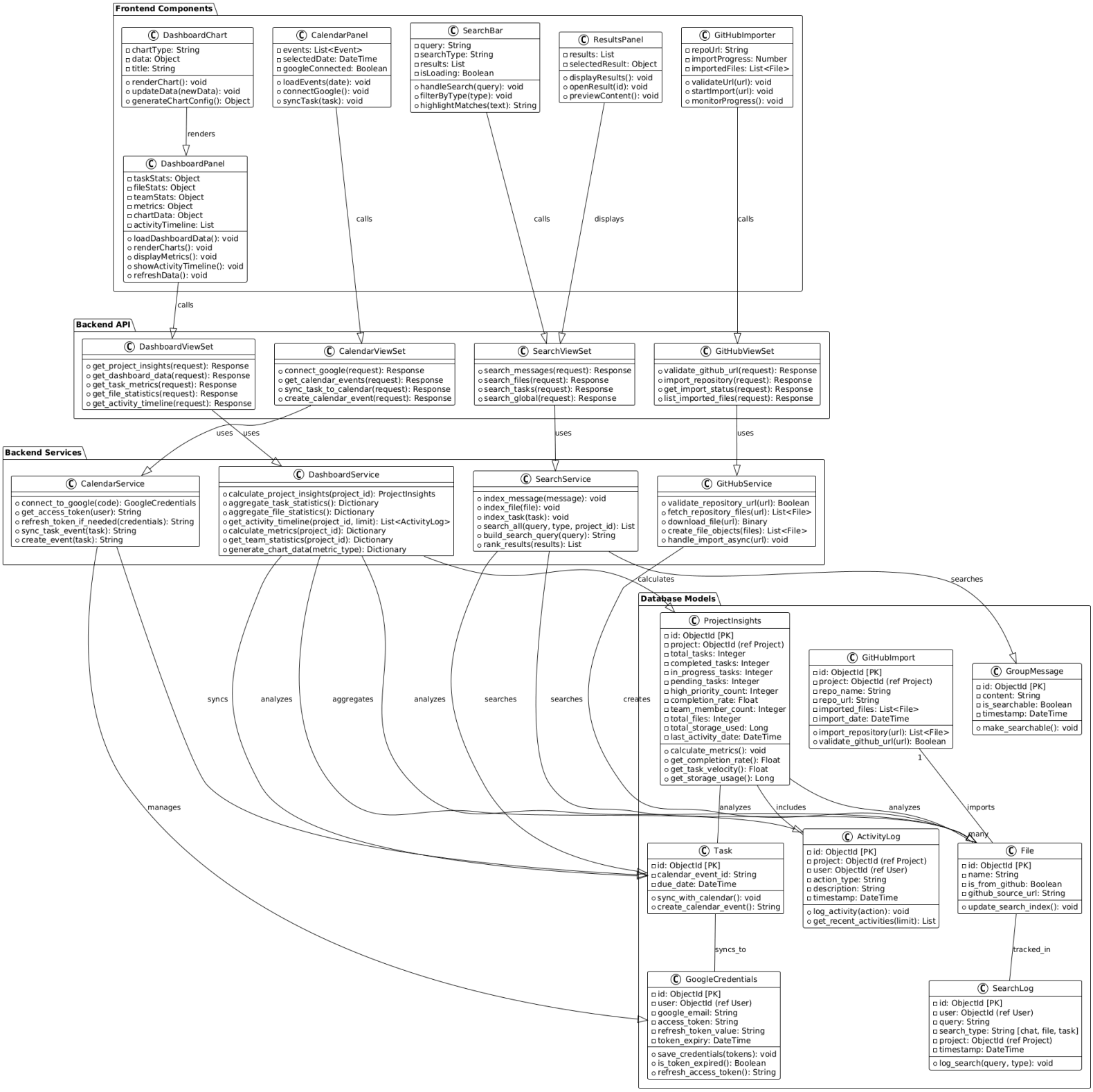

Browser
TaskDetailsPanel
DependencyModal
APIClient
Backend
Database
Add Task Dependency
1 View task details
2 Show dependencies section
3 Click "Add dependency"
4 Show available tasks modal
5 GET /tasks?project_id={id}&exclude={current_task_id}
6 Get available tasks
7 Query Project tasks
8 List of tasks
9 Exclude current task and completed tasks
10 Return task list
11 Display tasks
12 Select task to depend on
13 POST /tasks/{task_id}/add-dependency/ {depends_on_task_id}
14 Add dependency request
15 Authenticate user
16 Query both tasks
alt [Task Not Found]
17 One or both tasks not found
[Both Tasks Found]
18 Check for circular dependency
alt [Circular Dependency Detected]
19 Cannot add - would create circular reference
20 Show error message
[No Circular Dependency]
21 Save dependency relationship
22 Dependency created
23 Mark dependent task as blocked
24 Dependency added successfully
25 Refresh dependencies list
26 Show new dependency
27 Dependency added ✓
Browser
TaskDetailsPanel
DependencyModal
APIClient
Backend
Database


Browser
FileManager
ChatComponent
APIClient
Backend
S3Storage
Database
Download File
1 Click download in file manager
alt [File From File Manager]
2 GET /files/{file_id}/download
[File From Chat]
3 Click download in chat message
4 GET /files/{file_id}/download
5 Download file request
6 Authenticate user (verify project access)
7 Query File {s3_key}
8 File metadata
alt [File Not Found]
9 File not found (may be deleted)
10 Show error message
11 Show error message
[File Found]
12 Verify user has project access
alt [Access Denied]
13 Insufficient permissions
[Access Granted]
14 Generate temporary download URL (signed URL)
15 Temporary URL (expires in 15 min)
16 Return download URL
17 Provide download link
18 Provide download link
19 Start download
20 Start download
21 Download file data
22 File streamed to browser
23 File downloaded to device ✓
Browser
FileManager
ChatComponent
APIClient
Backend
S3Storage
Database

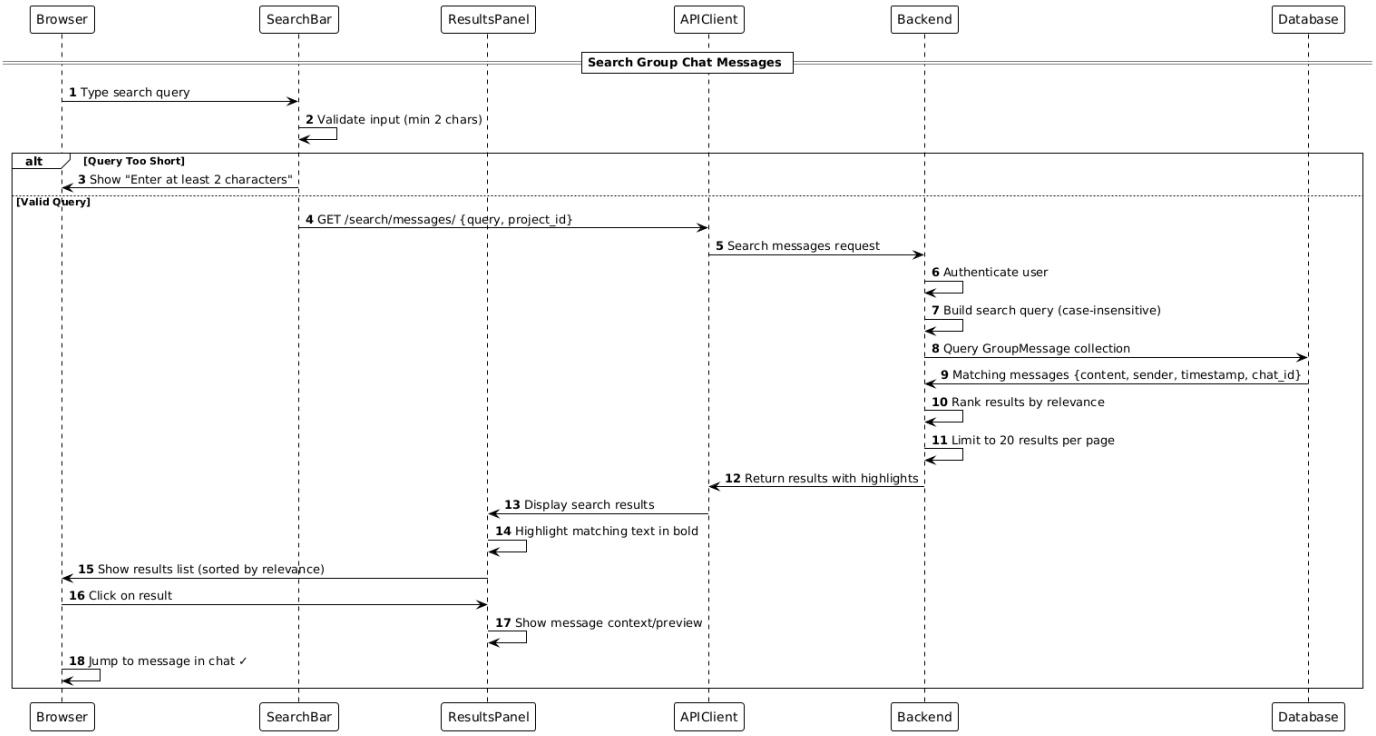
Sprint 4– Advanced Features

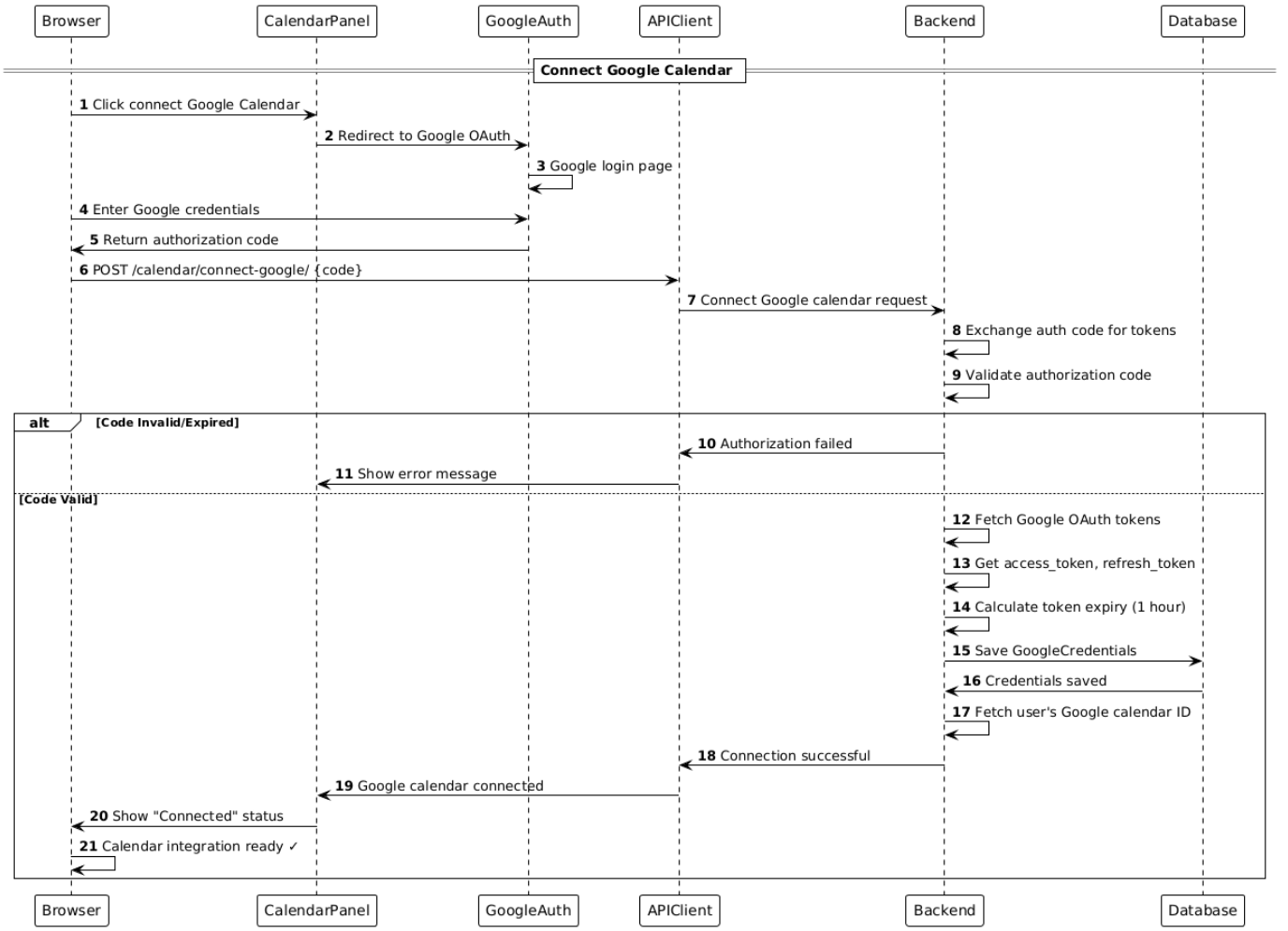
- **Sprint Goal:** Improve usability with search, integrations and project insights.
- **Duration:** 1.5 week.
- **Deliverables:**
  - Search functionality.
  - Third-party tool integrations.
  - Visual dashboard for project insights.
- **Key User Story:** US-6, US-8, US-12, US-18, US-19, US-20.

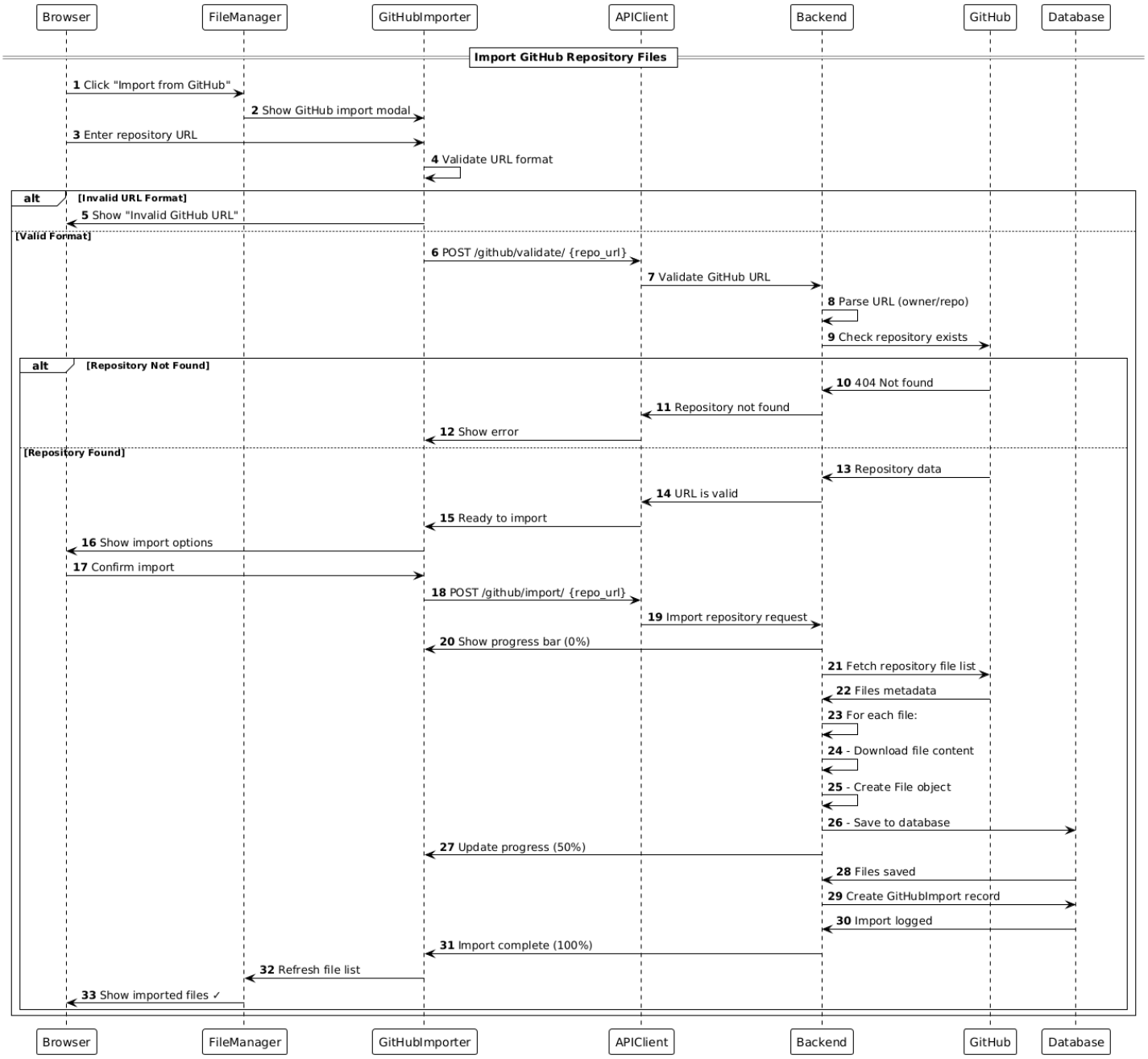
- ## Class Diagram:



- **Sequence Diagrams:**



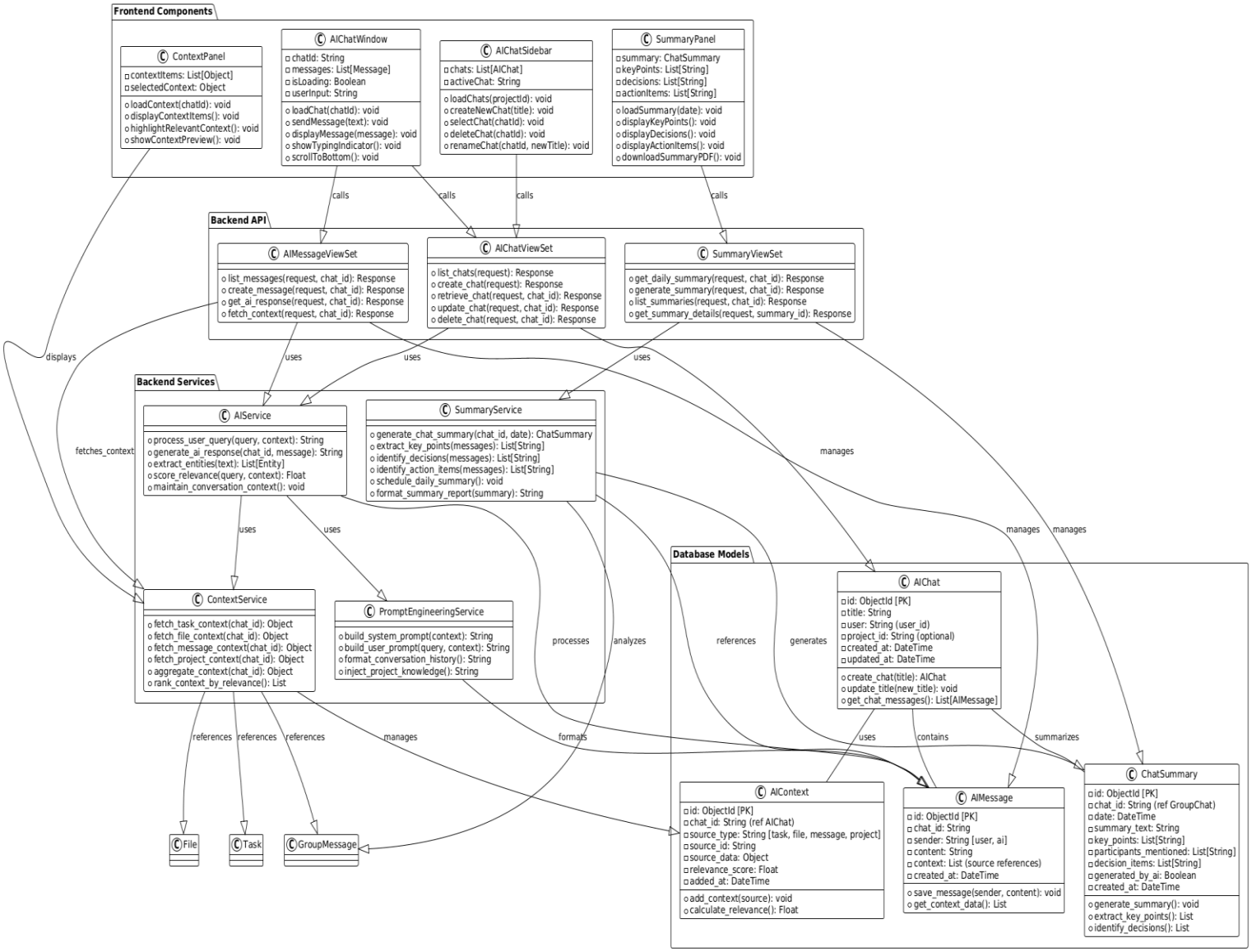



Browser
FileManager
GitHubImporter
APIClient
Backend
GitHub
Database
Import GitHub Repository Files
1 Click "Import from GitHub"
2 Show GitHub import modal
3 Enter repository URL
4 Validate URL format
alt
[Invalid URL Format]
5 Show "Invalid GitHub URL"
[Valid Format]
6 POST /github/validate/ {repo_url}
7 Validate GitHub URL
8 Parse URL (owner/repo)
9 Check repository exists
alt
[Repository Not Found]
10 404 Not found
11 Repository not found
12 Show error
[Repository Found]
13 Repository data
14 URL is valid
15 Ready to import
16 Show import options
17 Confirm import
18 POST /github/import/ {repo_url}
19 Import repository request
20 Show progress bar (0%)
21 Fetch repository file list
22 Files metadata
23 For each file:
24 - Download file content
25 - Create File object
26 - Save to database
27 Update progress (50%)
28 Files saved
29 Create GitHubImport record
30 Import logged
31 Import complete (100%)
32 Refresh file list
33 Show imported files ✓

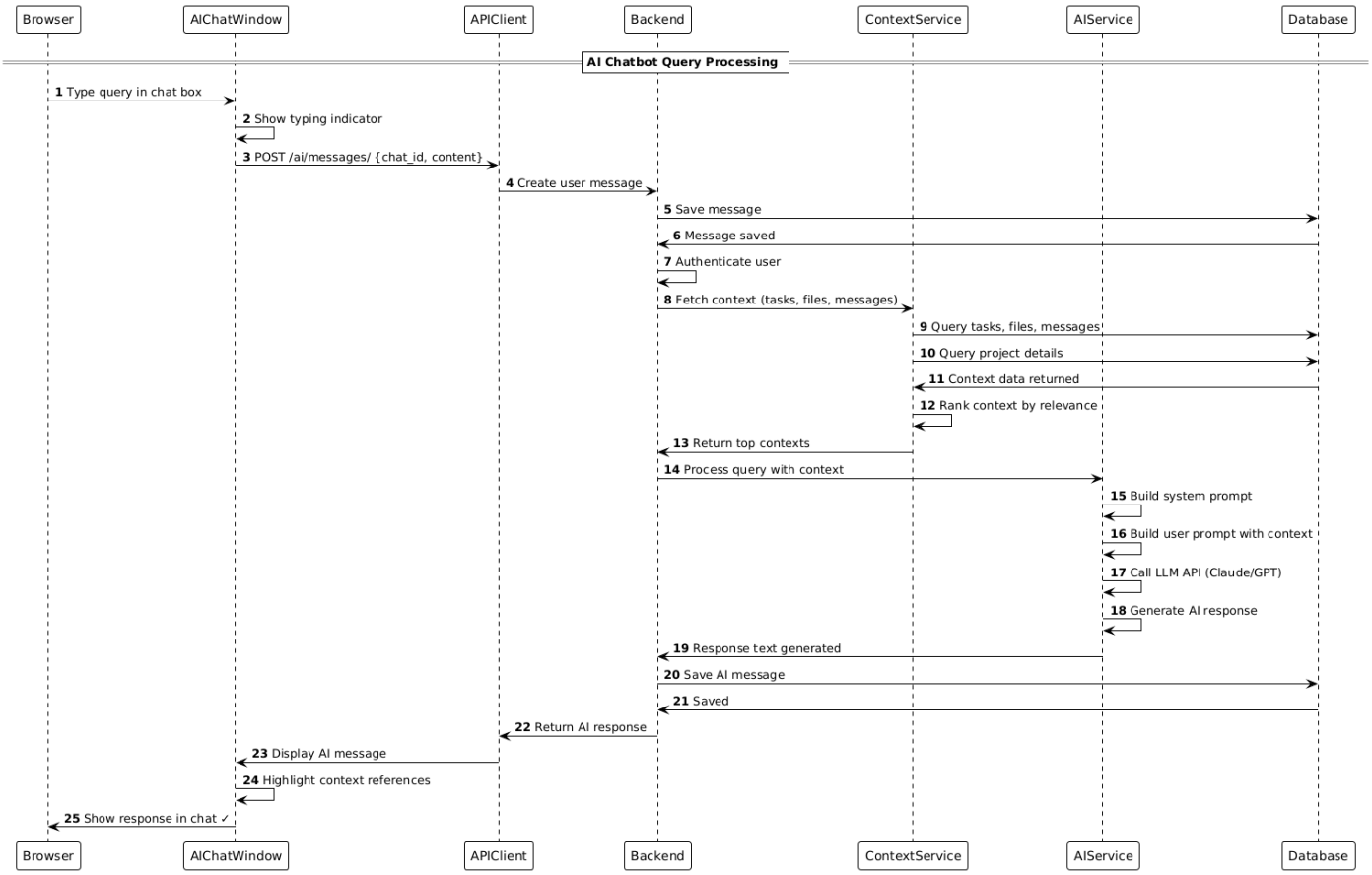
Sprint 5– AI Assistance

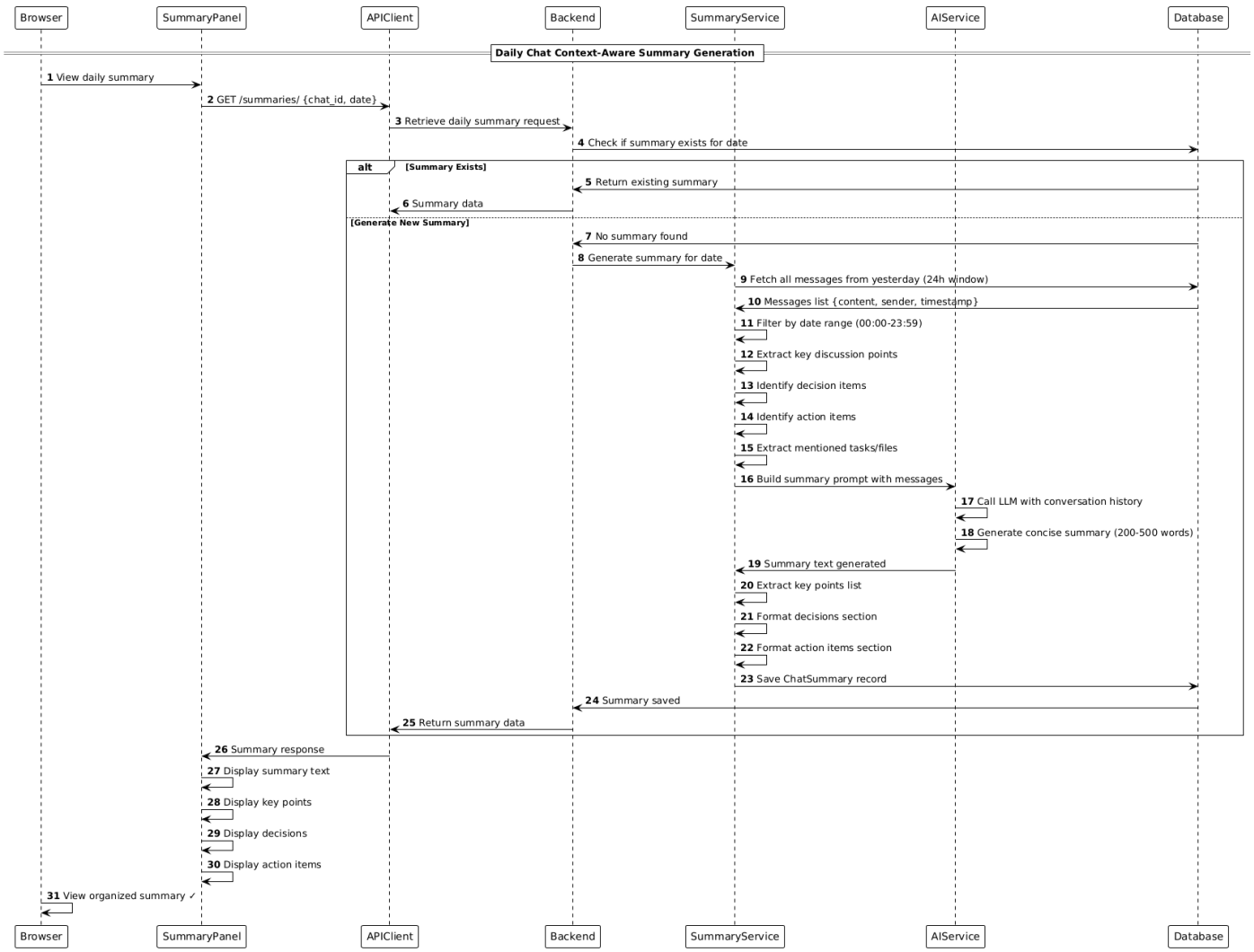
- **Sprint Goal:** Provide AI driven features to save time.
- **Duration:** 1 week.
- **Deliverables:**
  - AI chatbot for user queries.
  - Daily chat context-aware summaries.
- **Key User Story:** US-11.

- ## Class Diagram:



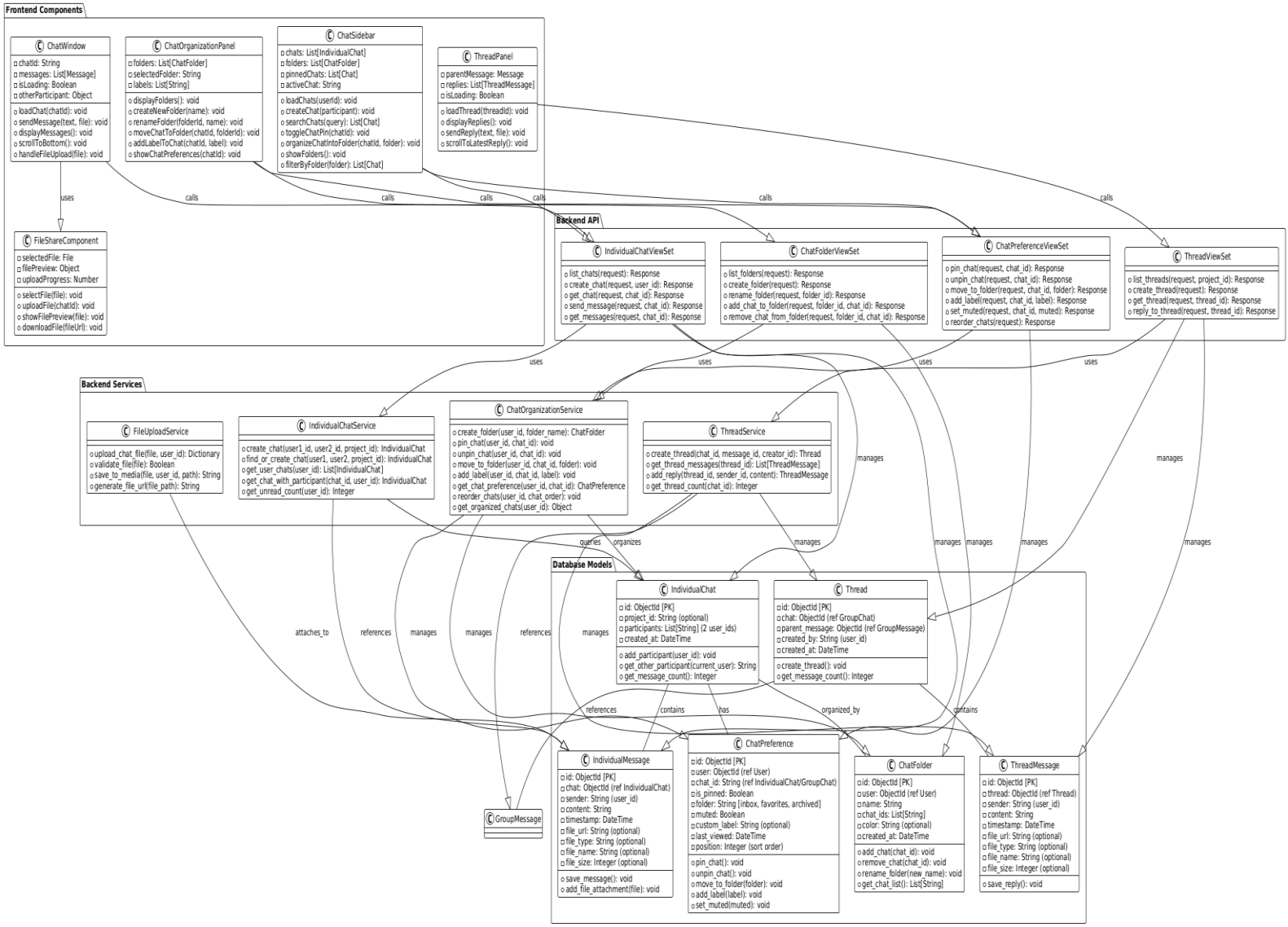
- **Sequence Diagrams:**



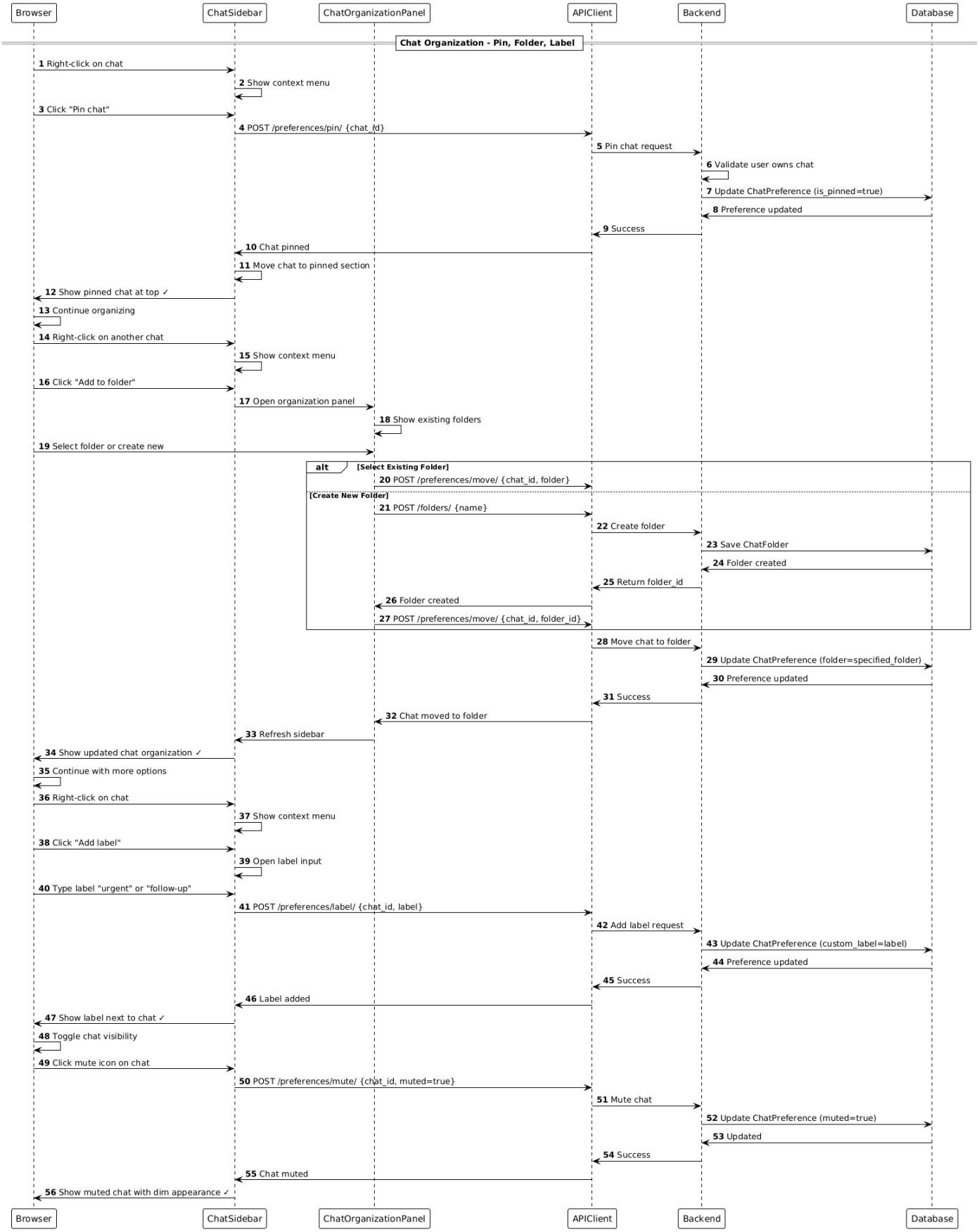


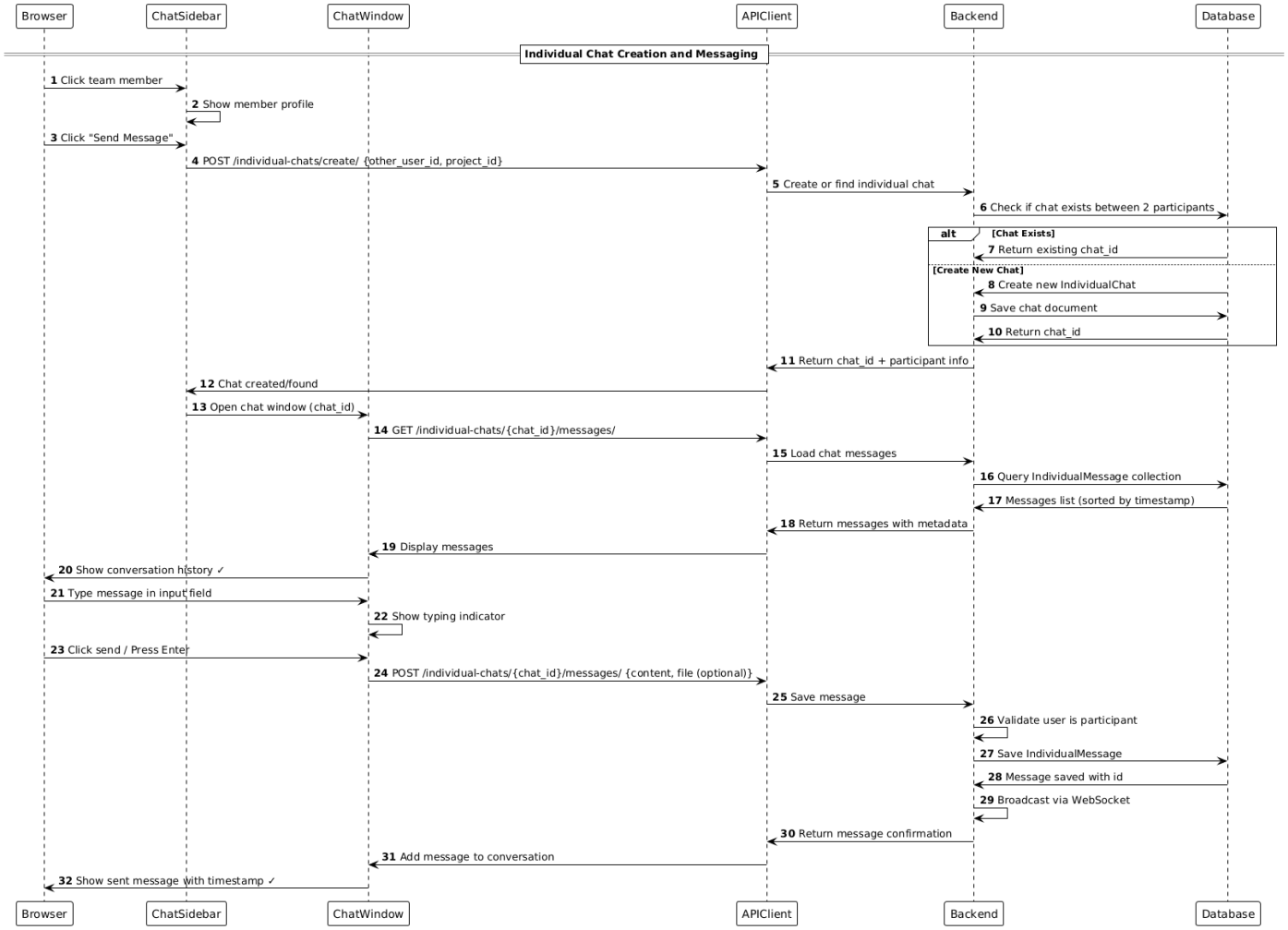
- # Sprint 6 – Custom Collaboration & Version Control
- **Sprint Goal:** Personalize collaboration and improve version management.
- **Duration:** 1 week.
- **Deliverables:**
    - Individual chat.
    - Organizable chats.
    - Version control and rollback to previous versions.
- **Key Us r Story:** US-9, US-14.

- ## Class Diagram:



- **Sequence Diagram:**




Browser
ChatSidebar
ChatWindow
APIClient
Backend
Database
Individual Chat Creation and Messaging
1 Click team member
2 Show member profile
3 Click "Send Message"
4 POST /individual-chats/create/ {other_user_id, project_id}
5 Create or find individual chat
6 Check if chat exists between 2 participants
alt
[Chat Exists]
7 Return existing chat_id
[Create New Chat]
8 Create new IndividualChat
9 Save chat document
10 Return chat_id
11 Return chat_id + participant info
12 Chat created/found
13 Open chat window (chat_id)
14 GET /individual-chats/{chat_id}/messages/
15 Load chat messages
16 Query IndividualMessage collection
17 Messages list (sorted by timestamp)
18 Return messages with metadata
19 Display messages
20 Show conversation history ✓
21 Type message in input field
22 Show typing indicator
23 Click send / Press Enter
24 POST /individual-chats/{chat_id}/messages/ {content, file (optional)}
25 Save message
26 Validate user is participant
27 Save IndividualMessage
28 Message saved with id
29 Broadcast via WebSocket
30 Return message confirmation
31 Add message to conversation
32 Show sent message with timestamp ✓
Browser
ChatSidebar
ChatWindow
APIClient
Backend
Database