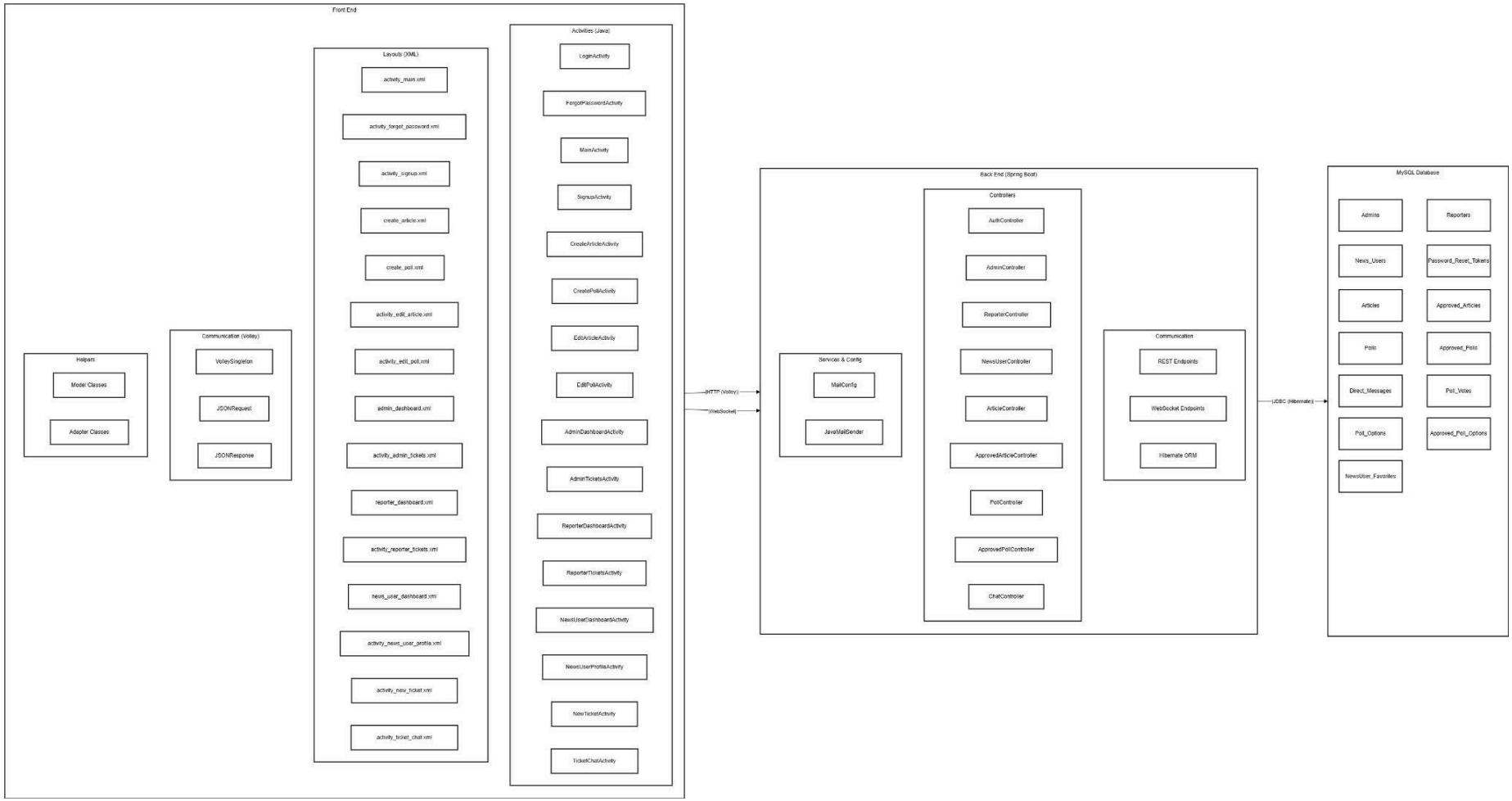
Design Document for PollyPress

Group 1_mahdi_3

Vikrant Gandotra: 50% contribution

M Shivdhar Reddy: 50% contribution


Front End
Activities (Java)
LoginActivity
ForgotPasswordActivity
MainActivity
SignupActivity
CreateArticleActivity
CreatePollActivity
EditArticleActivity
EditPollActivity
AdminDashboardActivity
AdminTicketsActivity
ReporterDashboardActivity
ReporterTicketsActivity
NewsUserDashboardActivity
NewsUserProfileActivity
NewTicketActivity
TicketChatActivity
Layouts (XML)
activity_main.xml
activity_forgot_password.xml
activity_signup.xml
create_article.xml
create_poll.xml
activity_edit_article.xml
activity_edit_poll.xml
admin_dashboard.xml
activity_admin_tickets.xml
reporter_dashboard.xml
activity_reporter_tickets.xml
news_user_dashboard.xml
activity_news_user_profile.xml
activity_new_ticket.xml
activity_ticket_chat.xml
Communication (Volley)
VolleySingleton
JSONRequest
JSONResponse
Helpers
Model Classes
Adapter Classes
(HTTP (Volley))
(WebSocket)
Back End (Spring Boot)
Controllers
AuthController
AdminController
ReporterController
NewsUserController
ArticleController
ApprovedArticleController
PollController
ApprovedPollController
ChatController
Services & Config
MailConfig
JavaMailSender
Communication
REST Endpoints
WebSocket Endpoints
Hibernate ORM
(JDBC (Hibernate))
MySQL Database
Admins
Reporters
News_Users
Password_Reset_Tokens
Articles
Approved_Articles
Polls
Approved_Polls
Direct_Messages
Poll_Votes
Poll_Options
Approved_Poll_Options
NewsUser_Favorites

Design Description

1. Front End

Entry & Navigation

- *MainActivity (Login) is the gateway: users choose their role, enter credentials, or navigate to SignupActivity or ForgotPasswordActivity.*

- *Once authenticated, everyone lands on their role-specific Dashboard (Admin, Reporter, or NewsUser), from which all other flows branch.*

UI Modules

- *Authentication:*

    - *Login, Sign-Up, and Forgot Password screens handle user onboarding and credential recovery.*

- *Dashboards:*

    - *AdminDashboardActivity: Admins see two side-by-side lists—Unapproved Articles and Pending Polls—where each item offers Approve, Edit, and Delete buttons. Below these lists is a Tickets button that, when tapped, takes the admin to a screen showing every ticket reporters have submitted.*

    - *ReporterDashboardActivity: Reporters land on a view with two sections—My Articles and My Polls—each card displaying Edit and Delete actions. Floating action buttons let them create a new article or poll. A My Tickets button opens the ticket list, where reporters can both view all tickets they've sent to admins and create new ones.*

    - *NewsUserDashboardActivity: News users switch between two tabs—Articles and Polls. The Articles tab shows a scrollable list of approved news items; the Polls tab displays each poll along with the percentage of votes cast by others. Both tabs support infinite scrolling and let users dive into full details or cast a vote.*

- *Profile & Settings:*

    - *NewsUserProfileActivity (and the shared Edit-Profile dialog) let users update their name, email, or picture, share the app, log out, or delete their account.*

- *Content Management (Reporters):*

    - *Create/Edit Article and Create/Edit Poll screens provide form fields, option-builders, and submit/update/delete actions.*

- *Ticketing & Chat:*

    - *NewTicketActivity collects a subject and message.*

    - *TicketListActivity (per role) lists existing tickets, shows assignment controls for reporters, and contextual "Chat"/"Resolve" actions for admins.*

    - *TicketChatActivity displays a live chat thread under each ticket, letting users send and receive messages in real time.*

Supporting Layers

- *Communication is handled by a VolleySingleton for all REST calls and by WebSocket managers for live poll-vote and ticket-chat streams.*

- *Adapters bind JSON-backed Models (articles, polls, tickets, sessions) to RecyclerViews and ViewPagers, keeping the UI code clean and modular.*

2. Backend

Actors & Entry Point

Clients interact with our backend via HTTP and WebSocket. All REST calls arrive at one of nine controllers, each responsible for a distinct resource (admins, reporters, users, articles, polls) or function (authentication, chat, password reset). Real-time chat and polling flows use WebSocket endpoints to push and receive messages immediately.

Controllers

• AdminController (/admins) provides full CRUD on administrator accounts and links to approved articles.

• ReporterController (/reporters) provides full CRUD on reporter profiles.

• AuthController (/auth/login) validates credentials for news users, reporters, and admins, issuing JWTs for downstream security.

• NewsUserController (/newsusers) manages reader accounts plus a two-step password-reset flow: tokens are generated, emailed, and validated.

• ArticleController (/articles) handles article creation, edits, “pending deletion,” and approval; it moves approved items into ApprovedArticleController.

• ApprovedArticleController (/approved-articles) manages finalized articles with standard CRUD operations.

• ChatController (/chats) returns direct-message histories on demand.

• PollController (/polls) allows reporters to create and update polls.

• ApprovedPollController (/approved-polls) surfaces finalized polls after administrative approval.

Persistence Layer

Each controller delegates data access to a corresponding Spring Data JPA repository. Repositories interact with MySQL via Hibernate, mapping our ten entity classes—plus three collection/join tables—to their tables. The MailConfig bean and Spring's JavaMailSender handle password-reset emails.

Real-Time Modules

• ChatWebSocket maintains session maps and persists each DirectMessage in direct_messages; incoming JSON payloads are echoed to sender and recipient.

• PollSocket accepts "OPTION:" messages, records each vote in poll_votes, and broadcasts vote events to all connected clients.

Database Schema
Our schema consists of user tables (admins, reporters, news_users, password_reset_tokens), content tables (articles, approved_articles, polls, approved_polls), real-time tables (direct_messages, poll_votes), and supporting collection tables (poll_options, approved_poll_options, newsuser_favorites). JPA annotations define foreign-key relationships so that each entity's lifecycle is managed automatically.

Database Schema