Design Document for Cy's World

Group 2_swarna_4

Jawad: 25 % contribution

Kiishi: 25% contribution

Jayden: 25% contribution

Sonia Patil: 25% contribution

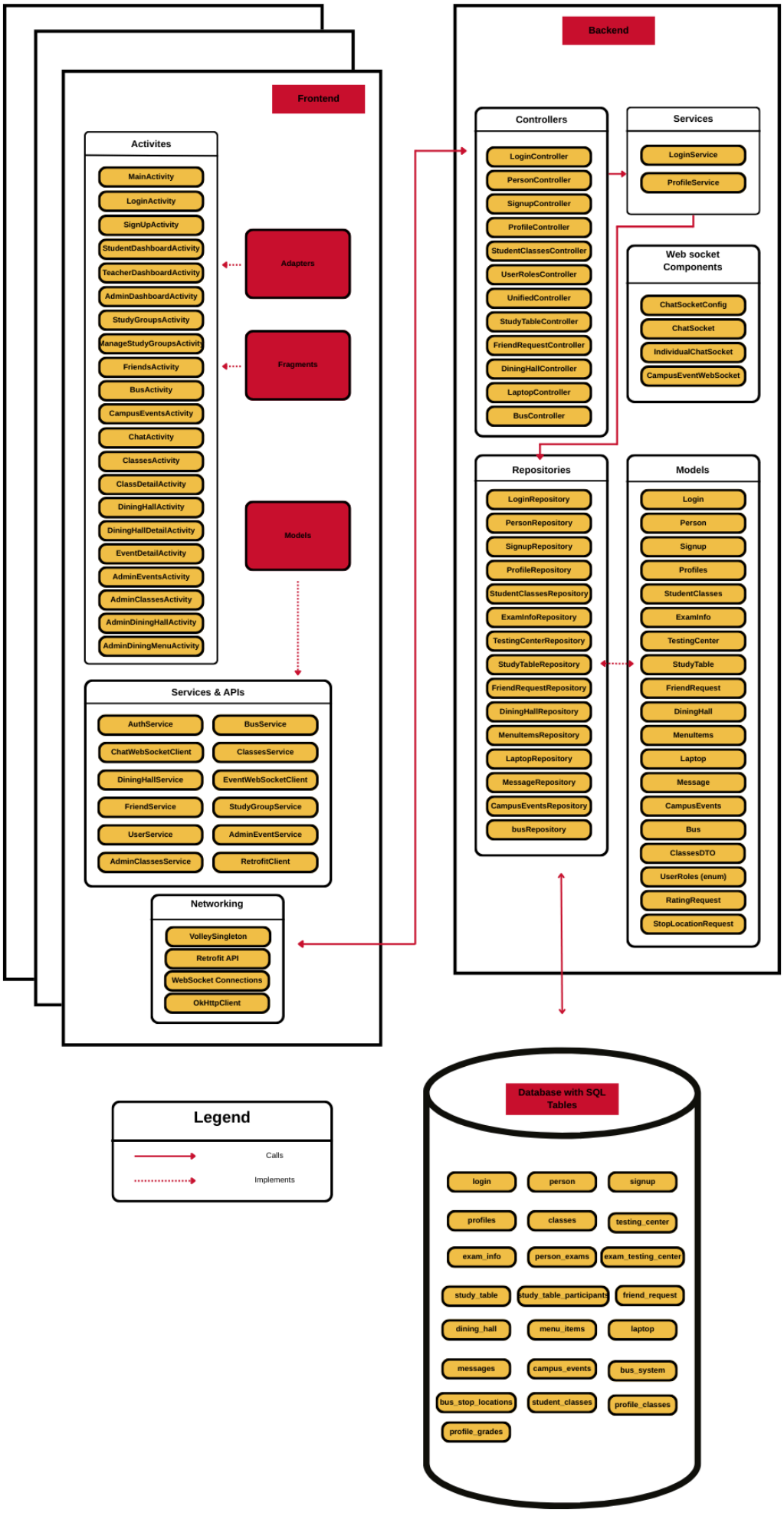
For later submission:

Swagger documentation link should be:

http://coms-3090-017.las.iastate.edu:8080/swagger-ui/index.html

we need to remake the jar file and run it I presume so that they can access it.

PUT THE BLOCK DIAGRAM PICTURE ON THIS PAGE! (Create the picture using pencil or drawIO)


Frontend
Activites
MainActivity
LoginActivity
SignUpActivity
StudentDashboardActivity
TeacherDashboardActivity
AdminDashboardActivity
StudyGroupsActivity
ManageStudyGroupsActivity
FriendsActivity
BusActivity
CampusEventsActivity
ChatActivity
ClassesActivity
ClassDetailActivity
DiningHallActivity
DiningHallDetailActivity
EventDetailActivity
AdminEventsActivity
AdminClassesActivity
AdminDiningHallActivity
AdminDiningMenuActivity
Adapters
Fragments
Models
Services & APIs
AuthService
BusService
ChatWebSocketClient
ClassesService
DiningHallService
EventWebSocketClient
FriendService
StudyGroupService
UserService
AdminEventService
AdminClassesService
RetrofitClient
Networking
VolleySingleton
Retrofit API
WebSocket Connections
OkHttpClient
Backend
Controllers
LoginController
PersonController
SignupController
ProfileController
StudentClassesController
UserRolesController
UnifiedController
StudyTableController
FriendRequestController
DiningHallController
LaptopController
BusController
Services
LoginService
ProfileService
Web socket Components
ChatSocketConfig
ChatSocket
IndividualChatSocket
CampusEventWebSocket
Repositories
LoginRepository
PersonRepository
SignupRepository
ProfileRepository
StudentClassesRepository
ExamInfoRepository
TestingCenterRepository
StudyTableRepository
FriendRequestRepository
DiningHallRepository
MenuItemsRepository
LaptopRepository
MessageRepository
CampusEventsRepository
busRepository
Models
Login
Person
Signup
Profiles
StudentClasses
ExamInfo
TestingCenter
StudyTable
FriendRequest
DiningHall
MenuItems
Laptop
Message
CampusEvents
Bus
ClassesDTO
UserRoles (enum)
RatingRequest
StopLocationRequest
Legend
Calls
Implements
Database with SQL Tables
login
person
signup
profiles
classes
testing_center
exam_info
person_exams
exam_testing_center
study_table
study_table_participants
friend_request
dining_hall
menu_items
laptop
messages
campus_events
bus_system
bus_stop_locations
student_classes
profile_classes
profile_grades

Use this third page to describe complex parts of your design.

CyWorld Design Description

CyWorld is a comprehensive campus application designed for Iowa State University students, teachers, and administrators. The system follows a client-server architecture with role-based access and modular components.

System Architecture Overview

The application is structured in three main tiers:

1. **Frontend**: User interfaces for students, teachers, and administrators
2. **Backend**: Controllers, services, and repositories handling business logic
3. **Database**: SQL-based data storage with relational tables

Key Modules and Functionalities

User Authentication & Profiles

- Secure login system with role-based access (student, teacher, administrator)
- User profile management with personal information and preferences
- Friend connection system for students with request/accept workflow

Academic Features

- **Class Management**: Course enrollment, schedules, and materials
- **Grade System**: Teachers can input/update grades in real-time
- **Study Groups**: Students can create and manage study groups with friends

Campus Resources

- **Dining Hall System**: Real-time menu updates and facility status
- **Bus Routes**: Live tracking of campus transportation with arrival predictions
- **Campus Events**: Calendar with RSVP functionality and notifications

Communication Tools

- **Chat Interface**: Real-time messaging between connected users
- **Notifications**: System and user-generated alerts for important updates

Backend System Design

The backend implements a RESTful API architecture with the following components:

- **Controllers**: Handle HTTP requests and manage session state
    - Post: Sends information about different objects that need to be added to the database
    - Get: Requests information from the database, usually with an ID for the specific info
    - Put: Edits information about a specific item. This allows updates without creating new objects
    - Delete: removes information or an object from the database based on the ID given
- **Services**: Implement business logic and data processing
- **Repositories**: Interface with database for CRUD operations
- **WebSocket Components**: Support real-time features like chat and notifications

Data Persistence

The database design follows a normalized relational model with tables for:

- User accounts and profiles
- Academic data (classes, grades)
- Campus resources (dining, events, transportation)
- Social connections (friends, study groups)
- Chat history and notifications

Non-Functional Properties

- **Scalability**: System designed to handle at least 5 concurrent users, expandable to campus-wide deployment
- **Performance**: UI responsiveness under 0.5 seconds, login operations under 5 seconds

- **Real-time Updates**: Push notifications for critical information changes
- **Responsive Design**: Consistent UI across various Android screen sizes
- **Security**: Role-based access control with data privacy protection

PUT THE TABLE RELATIONSHIPS DIAGRAM on this fourth page! (Create the picture using MySQLWorkbench)

Database schemas