[1]

COMP4960 – Software Engineering Architectural Design
SyncUp

TEAM NAME, ID: JTNF, PROJECT GROUP 2
DATE: 02-28-2024

Outline

- Introduction
- Requirements
- System Architecture
  - Architecture Pattern(s)
  - Overall Architecture
  - Components Mapping
- Technology Stack Selection

Introduction

Project: An application that people can upload data about their schedule and availability. Other people in one's personal social network can join the platform and provide their data as well which will be analyzed by the application to suggest recommended dates and times to establish event plans.

Characteristics:

- The application will allow people to be able to join virtual social networks/groups and see when others are available for potential plans
- The application will make recommendations to users with regards to date and time for events
- The application will create and save events into a digital calendar at the request of users

[2]

Requirements

Functional

2.1.1 Upon starting, the system shall be able to generate a new user account.

2.1.2. Upon starting, the system shall be able to login to an existing user account.

2.1.4. Once a user is logged in, the system shall display the logged in user's availability and any user's availability in the same groups. The system shall be able to display availability by the hour, day, week, month, or year.

2.1.6. The system shall have a tab that displays all the groups users are connected to.

2.1.8. Once logged in, the system shall allow users to set their availability. The system shall offer availability to be set as repeating (daily, weekly, monthly, yearly) or one-time.

2.1.9. Within a group, the system shall have a button for planning a time to meet.

2.1.10. Upon selecting the button 2.1.9, the system shall suggest times that every group member is available within a specified time frame. If there is no time window when everyone is available, then the system will suggest alternatives with as many members as possible.

2.1.12. The system shall allow users to send friend requests to other users.

2.1.15 The GUI shall display a friends list and a group list upon user request.

Non-Functional

2.2.1. The system shall store login data via database to be queried for comparison with user input in order to authenticate user.

2.2.2. The system shall remove user login data when a user deletes their account.

2.2.3. Upon account deletion, the system shall delete availability data and reflect that on the UI.

2.2.4. The system database will link user account data to groups as one-to-many mapping.

2.2.5. The system should provide real-time user calendar updates of the group the user is viewing or when viewing availability of another user.

2.2.6. The method for suggesting meeting times within a group will use a greedy activity selection algorithm.

2.2.7. The system shall be able to handle 1,000 requests/queries at once.

2.2.8. The system shall respond to any user input within 1 second.

2.2.9. API integration for backend and front-end interactions. Consecutive requests are needed to ensure any changes to data stored in the database are shown on UI.

2.2.10. The system shall be available to use and interact with 24/7, with maintenance exceptions.

Architectural Pattern

[3]

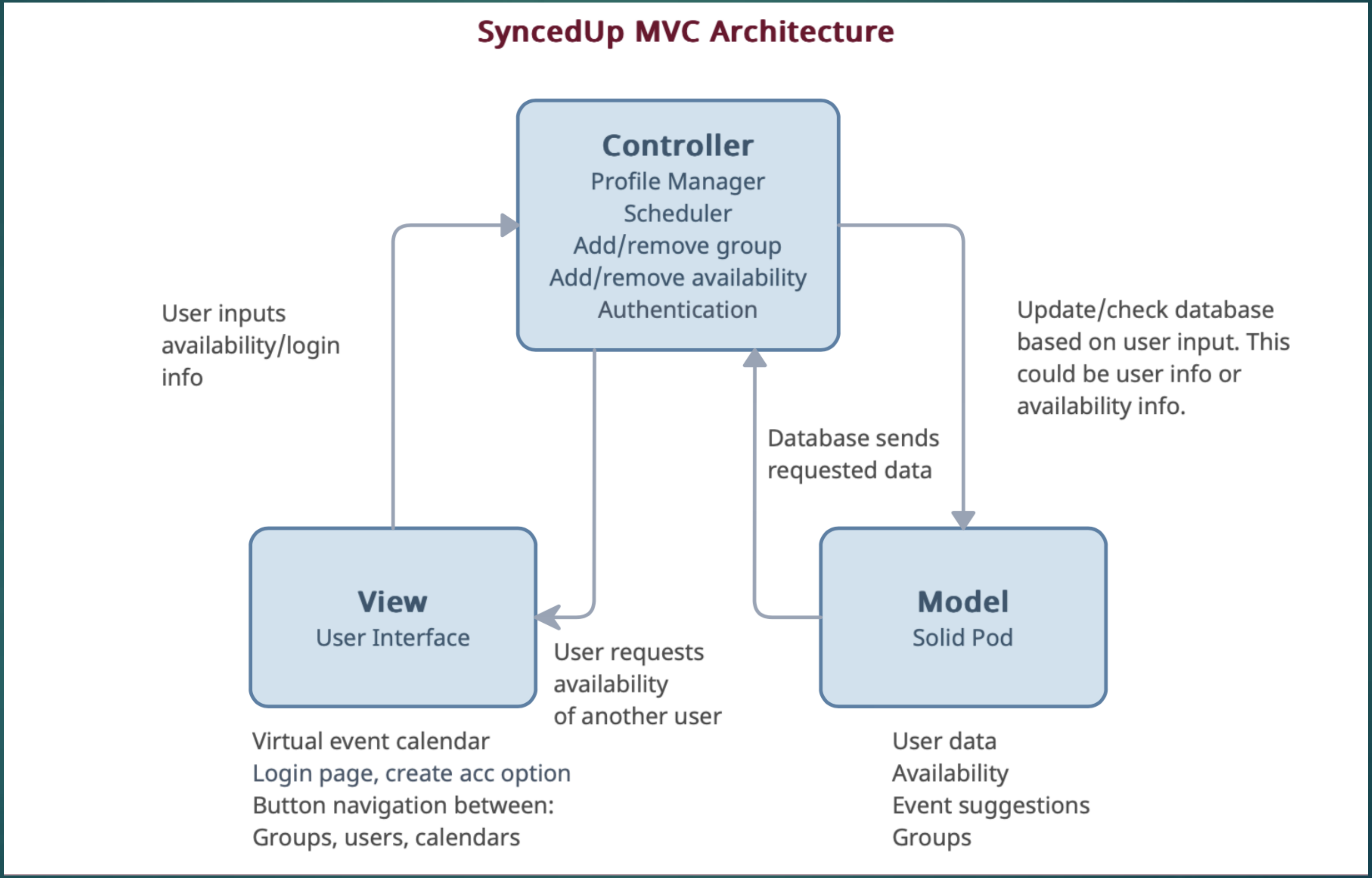
Model-View-Controller (MVC) architectural pattern will be used.

Because:
1) The calendar UI is our view, controller (or event handler), the model will hold data of user's availability and the controller will be used to update the model when events are added, deleted, etc.

2) Useful for user interaction and independence, which is implicit to web applications

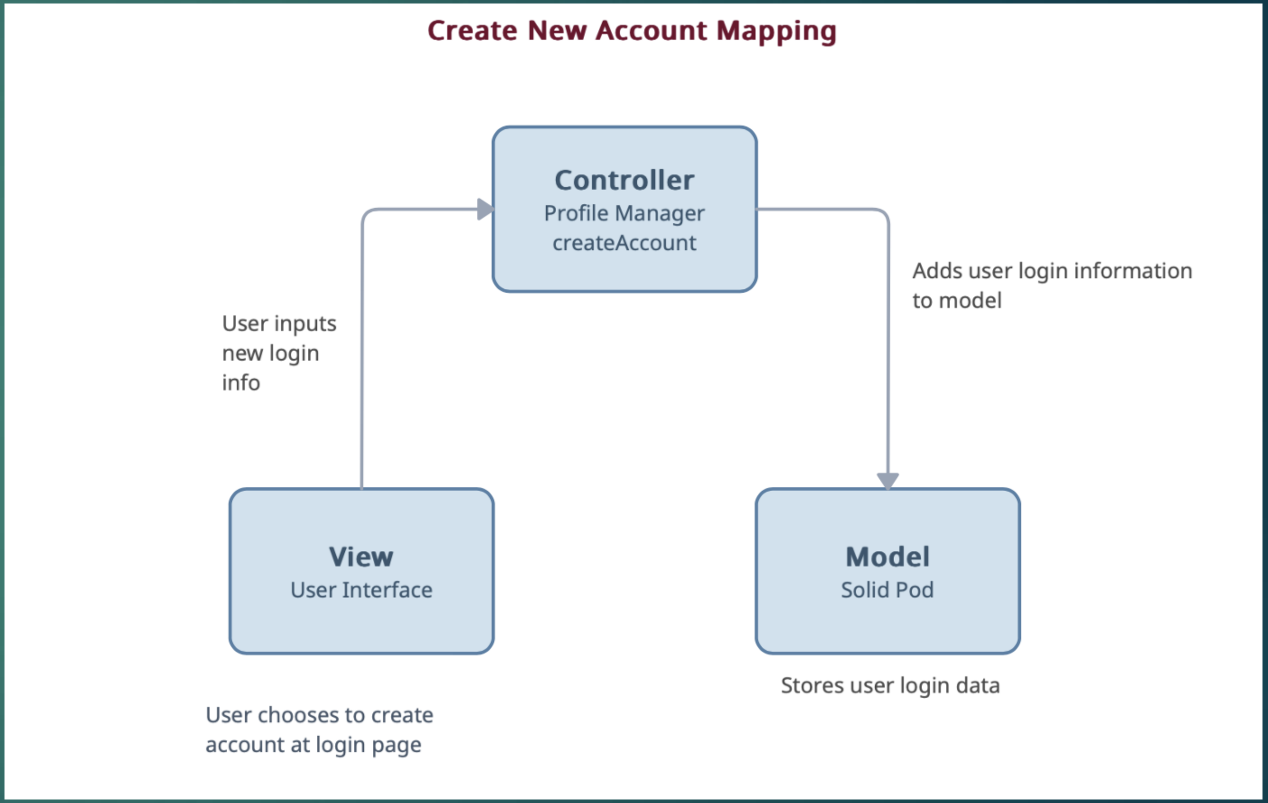
3) There are limited updates in data occurring (e.g. change in people or events) that we need to program, so even if this has the disadvantage of making the programming more complex, it's not major

Overall Architecture


SyncedUp MVC Architecture
Controller
Profile Manager
Scheduler
Add/remove group
Add/remove availability
Authentication
User inputs availability/login info
Update/check database based on user input. This could be user info or availability info.
Database sends requested data
User requests availability of another user
View
User Interface
Model
Solid Pod
Virtual event calendar
Login page, create acc option
Button navigation between:
Groups, users, calendars
User data
Availability
Event suggestions
Groups

Components Mapping for Functional Requirements

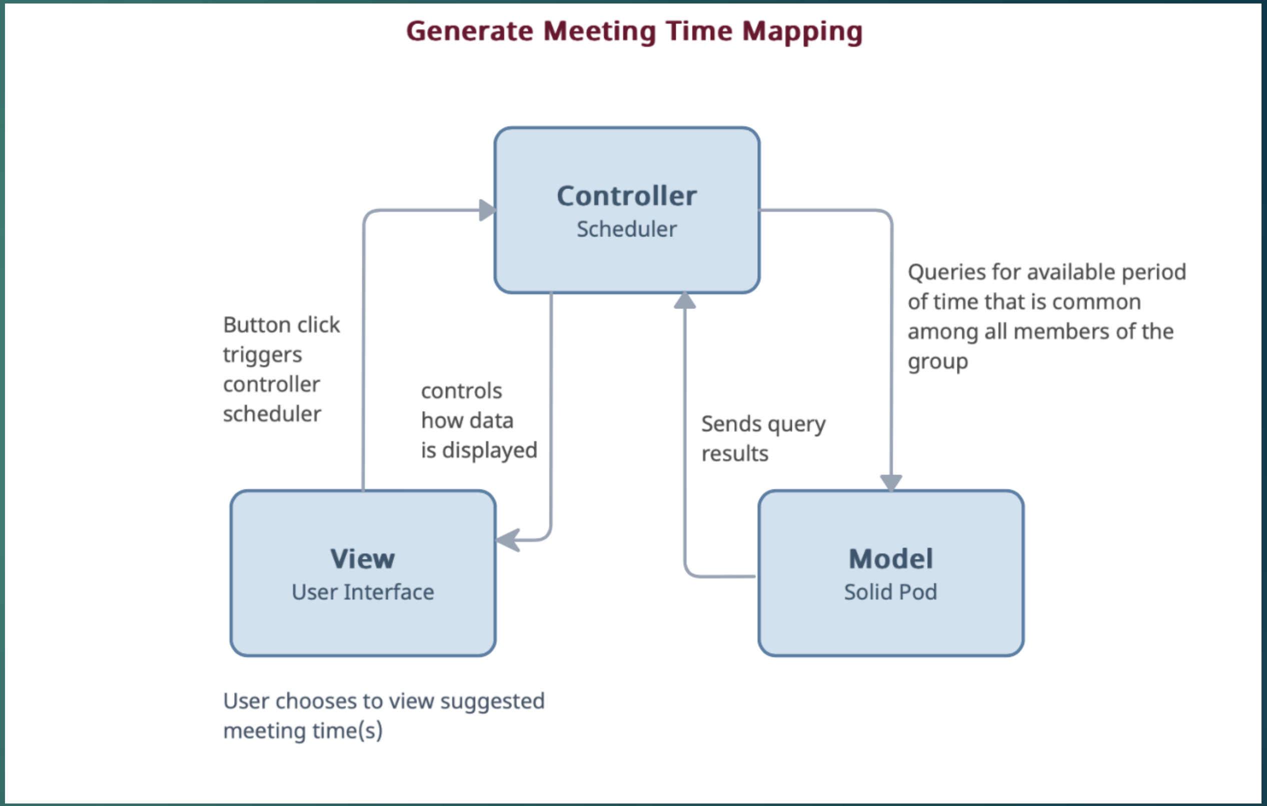
2.1.2. Upon starting, the system shall be able to generate a new user account.


Create New Account Mapping
Controller
Profile Manager
createAccount
User inputs new login info
Adds user login information to model
View
User Interface
Model
Solid Pod
Stores user login data
User chooses to create account at login page

Components Mapping for Functional Requirements

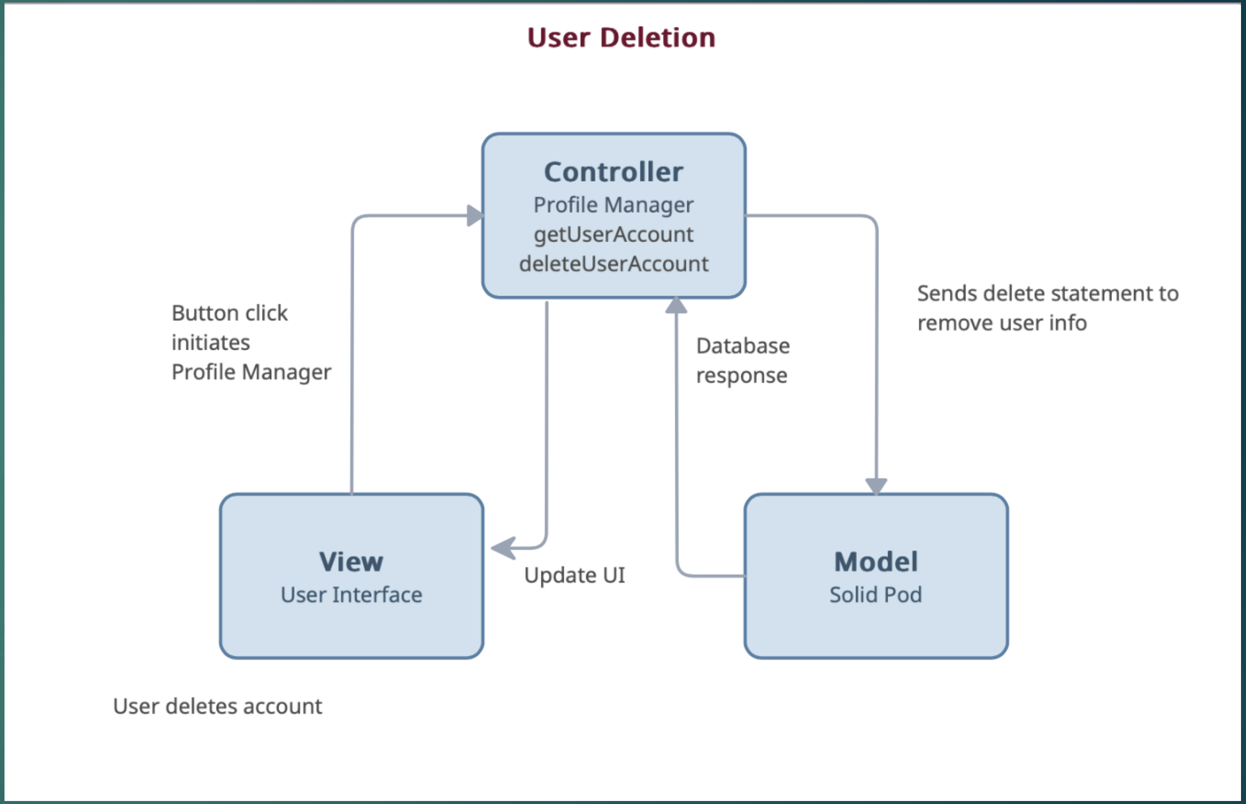
2.1.9. Within a group, the system shall have a button for planning a time to meet.

2.1.10. Upon selecting the button 2.1.9, the system shall suggest times that every group member is available within a specified time frame. If there is no time window when everyone is available, then the system will suggest alternatives with as many members as possible.


Generate Meeting Time Mapping
Controller
Scheduler
Button click triggers controller scheduler
controls how data is displayed
Sends query results
Queries for available period of time that is common among all members of the group
View
User Interface
Model
Solid Pod
User chooses to view suggested meeting time(s)

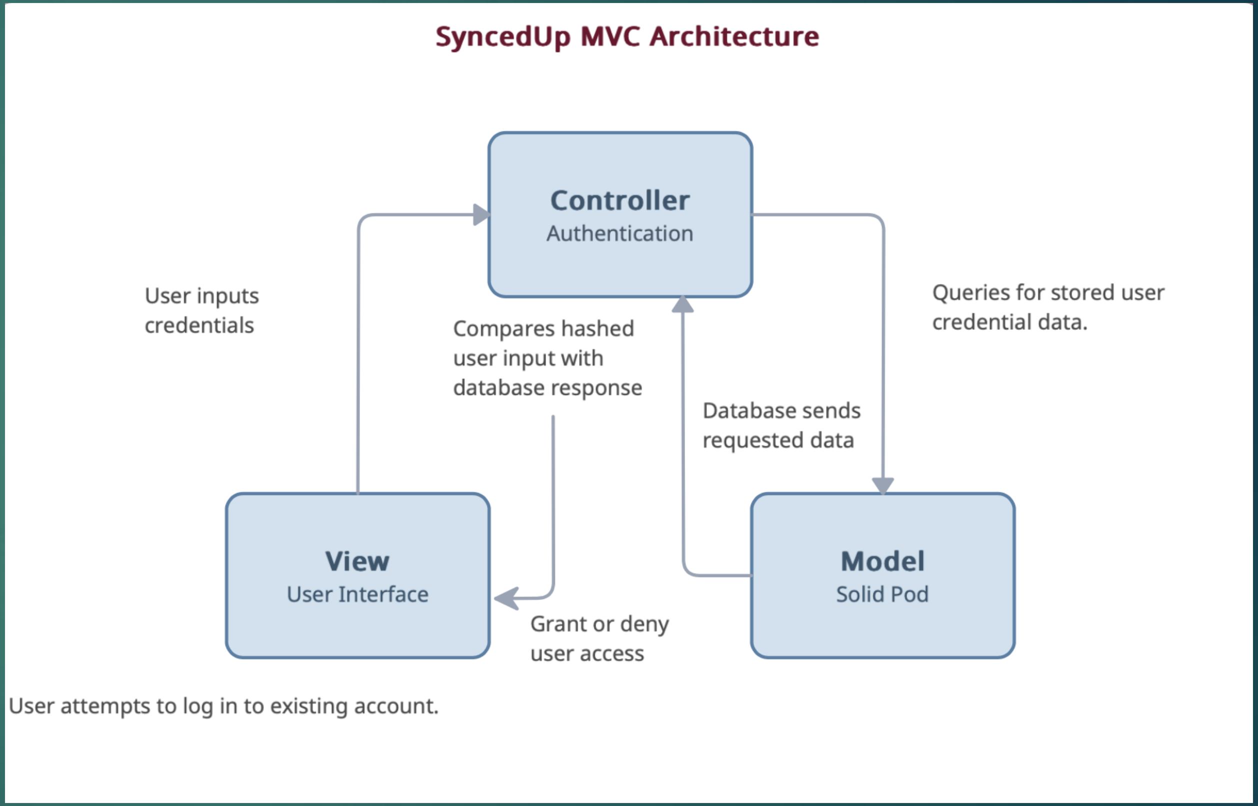
Components Mapping for Functional Requirements

2.2.3 Upon account deletion, the system shall delete availability data and reflect that on the UI.



Components Mapping for Non-Functional Requirements

2.2.1 The system shall store login data via database to be queried for comparison with user input to authenticate user


SyncedUp MVC Architecture
Controller
Authentication
User inputs credentials
Compares hashed user input with database response
Queries for stored user credential data.
Database sends requested data
View
User Interface
Grant or deny user access
Model
Solid Pod
User attempts to log in to existing account.

Technology Stack Selection

- *Solid: framework designed for decentralized web with users having control over personal data; different applications can use and retrieve same data as it is separated from a single entity; personal online data stores (Pods) store data and share to applications as needed*
- *WebID: Uniform Resource Identifier (URI) used in Solid to identify a unique user; used in authentication and authorization when it comes to manipulating data*
- *API: HTTP Requests; GET; POST; PUT; DELETE; used to access data from Solid Pods*
- *JavaFX: framework for building UI for Java applications; tools for creating functional and appealing interfaces*

[4]

References

- *[1] Calendar free icon* [Photograph]. IconMarketPK. https://www.flaticon.com/free-icon/calendar_9926396
- [2] (2018). *Event Management* [Photograph]. Gem Designs. https://thenounproject.com/icon/event-management-1696329/
- [3] (2021). *Model-view-controller* [Photograph]. Ary Prasetyo. https://thenounproject.com/icon/model-view-controller-4199124/
- *[4] MIT's Solid project logo* [Photograph]. MIT. https://solid.mit.edu/