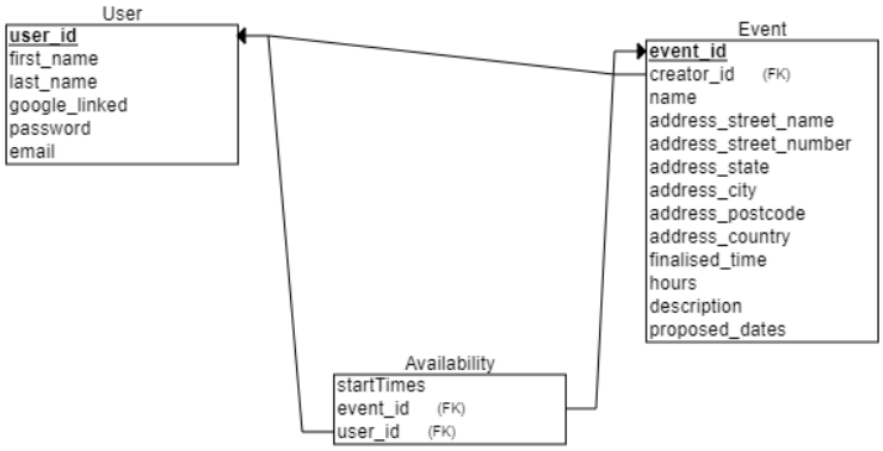
**WDC Data Plan : Patrick Sumarli, Nilesh Ramgolam, Ti Wei Lu, Muhammad Sharjeel Nawaz (Nilesh and friends)**

**Basic summary:**

- Users create events in the events table, they will also pick dates where they propose the event to be held on
- Users schedule their availability for events by picking timestamps on those proposed dates and storing them as an array
- When choosing their availability, we query the availability table to find all possible times and find out what times they aren't available
- Similarly for viewing their schedule
- Event creator can then pick a finalised time
- If everyone can join, the event is scheduled to everyone's calendar
- Otherwise, cancel the event or everyone re-schedule
- System admins can basically do anything on the web app

**Database Schema:**




**Sign up/log in.**

Where does the information come from?

- Pick user or sys_manager then login or sign up then to events page
- User access these routes which return html pages with forms
- Upon submission, a POST request will be sent to the same route
- GET & POST /signup
- GET & POST /login
- For login, the user will provide a username or password
- For signup, they will also include a few more details

What form should it take?

- User input are strings passed through the request body

If the information is on the server, what will the client need to send to retrieve that data?

- After user input, we respond with whether or not they are successful through changes to the frontend

If the information is on the client, how will it be sent to the server?

- POST Request body from the form

Does all information need to be stored on the server?

- We use the database for all storage here
- For third party authentication, we don't store anything in out database as we simply need an OK from their servers
- Client side holds on to a session token to maintain login status

What processing needs to be done to make the data useful?

- We need to use Open ID API to make the authentication secure
- Also, relevant third party authentication APIs
- We use the inputs in the request body to provide the API with information using AJAX
- Response is in the form of success/failure to the server

**Manage their user information.**

Where does the information come from?

- We provide the stored user information from the server
- We return a page and a page with a form for editing with the current information in semi-transparent text
- Users can edit their own info
- But the user must be logged in to do all this
- GET & POST /user/:user:id/settings

What form should it take?

- It should be a bunch of strings sent through req body

If the information is on the server, what will the client need to send to retrieve that data?

- A session token that authenticates them

If the information is on the client, how will it be sent to the server?

- Similar to authentication, we send a bunch of strings as part of the request body

Does all information need to be stored on the server?

- We store them in our database

What processing needs to be done to make the data useful?

- Update data on the database
- Respond with success/failure, maybe using alert() in Javascript

**Create a new event**

Where does the information come from?

- User needs to be authenticated
- We have a page with a form and a date picker for event creation
- GET & POST /user/:userid/create_event

What form should it take?

- It should be a bunch of strings like name, description, address, duration, etc
- User will also pass in an array of proposed dates for the event

If the information is on the server, what will the client need to send to retrieve that data?

- The client needs to be authenticated to make a request to this form

If the information is on the client, how will it be sent to the server?

- User sends it through a POST request body

Does all information need to be stored on the server?

- No

What processing needs to be done to make the data useful?

- Create a new data on the database
- Respond with success/failure then redirect them to the events page

**Generate a link for people without accounts.**

Where does the information come from?

- Event creator will be able to access this page if they are authenticated and view the events they created or attending
- GET /user/:userid/events
- Event creator can then make a post request to some button to get a string by making this GET request
- GET /user/:userid/get_link/:eventid
- Users need not be authenticated to schedule themselves for this event
- GET & POST /events/:eventid

What form should it take?

- We display a HTML page for the user's created events
- We generate a link for inviting users
- We display a HTML page for the event

If the information is on the server, what will the client need to send to retrieve that data?

- Nothing, just make the GET request since this is available for everyone regardless of authentication

If the information is on the client, how will it be sent to the server?

- Session token to authenticate event creator

Does all information need to be stored on the server?

- No

What processing needs to be done to make the data useful?

- Respond with success/failure then display the link to the client in the frontend

**Specify their availability for an event.**

Where does the information come from?

- GET & POST /events/:eventid/availability
- This should return a calendar-like form that pre-fills with their schedule and the proposed dates for the event

What form should it take?

- We should record a timestamp array of startTimes outlining when they are available, the ending time will be determined by the event duration itself

If the information is on the server, what will the client need to send to retrieve that data?

- Nothing

If the information is on the client, how will it be sent to the server?

- The client's selection will be sent as an array of timestamps via POST request body

Does all information need to be stored on the server?

- No, straight to the database

What processing needs to be done to make the data useful?

- Update the event-availability table in the database
- Respond with success/failure

**Link their calendar to automatically check their availability.**

Where does the information come from?

- GET /user/:userid/availability
- This route returns all the timestamps which the calendar will then process and display in the frontend in the proposed dates

What form should it take?

- An array of timestamps

If the information is on the server, what will the client need to send to retrieve that data?

- Authentication session token

If the information is on the client, how will it be sent to the server?

- N/A

Does all information need to be stored on the server?

- No

What processing needs to be done to make the data useful?

- We need to update the calendar to display when a user is not available, showing this on the frontend to the client

**See times when everyone is available for an event.**

Where does the information come from?

- Should be from the user-event table
- User needs to be authenticated for this
- GET /user/:userid/events/:eventid/availability
- This will also have a form for finalising the event time

What form should it take?

- Display timestamps for the proposed dates that the users have selected

If the information is on the server, what will the client need to send to retrieve that data?

- Authentication session token

If the information is on the client, how will it be sent to the server?

- N/A

Does all information need to be stored on the server?

- No

What processing needs to be done to make the data useful?

- Use the timestamp arrays to identify when all attendees overlap
- Respond with success/failure then show the completed event page to the creator

**Confirm/finalise an event time.**

Where does the information come from?

- POST /user/:userid/events/:eventid/availability
- The page to access this is in the previous feature

What form should it take?

- A finalised timestamp

If the information is on the server, what will the client need to send to retrieve that data?

- Authentication session token

If the information is on the client, how will it be sent to the server?

- POST request body

Does all information need to be stored on the server?

- No

What processing needs to be done to make the data useful?

- Update the database for a finalised time
- Check to see if everyone can make it, if not we ask the creator to either empty the timestamp array and reschedule or cancel the event
- Respond with success/failure and redirecting to the completed event page if successful

**Add the finalised event to their calendar.**

Where does the information come from?

- It is automatic. Users will be able to access this page if they are authenticated and view the events they created or attending
- GET /user/:userid/events
- Users will be able to automatically view their updated schedule by accessing this page to view a complete calendar
- GET /user/:userid/calendar

What form should it take?

- An HTML page with the calendar

If the information is on the server, what will the client need to send to retrieve that data?

- Authentication session token

If the information is on the client, how will it be sent to the server?

- N/A

Does all information need to be stored on the server?

- N/A

What processing needs to be done to make the data useful?

- The calendar will display which blocks of time are unavailable and what event exists there
- Respond with success/failure and redirecting to the updated calendar

## System Admins : can do everything

**Manage their user information.**

Where does the information come from?

- GET /admin/:adminid/info
- Will return a page showing all of the admin's user info
- GET & POST /admin/:adminid/info/edit
- Will return a page with a form for managing their user info

What form should it take?
- HTML pages with forms

If the information is on the server, what will the client need to send to retrieve that data?
- Admin authentication session token

If the information is on the client, how will it be sent to the server?
- POST request body

Does all information need to be stored on the server?
- No, just update the database

What processing needs to be done to make the data useful?
- Take the request body and update the database accordingly
- Respond with success/failure and show them the new user info

**Manage Users.**

Where does the information come from?
- GET /admin/:adminid/users
- Will display a list of users
- GET & POST /admin/:adminid/users/:userid
- Displays the user's information as a form to change

What form should it take?
- HTML pages with forms

If the information is on the server, what will the client need to send to retrieve that data?
- Admin authentication session token

If the information is on the client, how will it be sent to the server?
- POST request body

Does all information need to be stored on the server?
- No, just update the database

What processing needs to be done to make the data useful?
- Take the request body and update the database
- Respond with success/failure and show them the new user info

**Manage Events.**

Where does the information come from?
- GET /admin/:adminid/events
- Will display a list of events
- GET & POST /admin/:adminid/events/:eventid
- Displays the event's information as a form to change

What form should it take?
- HTML page with forms

If the information is on the server, what will the client need to send to retrieve that data?
- Admin authentication session token

If the information is on the client, how will it be sent to the server?

- POST request body

Does all information need to be stored on the server?

- No

What processing needs to be done to make the data useful?

- Take the request body and update the database
- Respond with success/failure and show them the new event info

**Sign-up other Admins.**

Where does the information come from?

- GET & POST admin/:adminid/signup
- Displays a form to create another admin account

What form should it take?

- HTML forms

If the information is on the server, what will the client need to send to retrieve that data?

- Admin authentication session token

If the information is on the client, how will it be sent to the server?

- POST request body

Does all information need to be stored on the server?

- No

What processing needs to be done to make the data useful?

- Take the request body and update the database
- Respond with success/failure by using alert()

**Users should be able to choose to link a social media/email/other account, allowing login via that platform, to make logging in easier.**

Where does the information come from?

- Existing API from those platforms
- Users will be able to make a POST request by clicking on buttons corresponding to the platform
- POST /signup/platform_name
- POST /login/platform_name
- They can link them through the user information page and signup for them

What form should it take?

- Buttons on the signup, login, and user information page

If the information is on the server, what will the client need to send to retrieve that data?

- Their credentials or details for the platform
- Proper authentication token from the platform

If the information is on the client, how will it be sent to the server?

- POST request

Does all information need to be stored on the server?

- No

What processing needs to be done to make the data useful?

- Access the API using the client's data and they will send a token to the client upon success
- We receive the token from the client and we send it back to the API to retrieve the information
- Respond with success/failure and reflect these changes to the frontend

**Users without accounts should be able to specify their availability for an event provided**

Where does the information come from?
- Users need not be authenticated to schedule themselves for this event
- GET & POST /events/:eventid
- The page will have a form to input their timestamps and name

What form should it take?
- We display a HTML page for the event
- The client will send back a list of timestamps they have chosen

If the information is on the server, what will the client need to send to retrieve that data?
- Nothing

If the information is on the client, how will it be sent to the server?
- Via post request body to the server

Does all information need to be stored on the server?
- No, we store them on the database

What processing needs to be done to make the data useful?
- We need to create data for the availability
- For anonymous users, we set the user_id to -1 since user_id starts from 0, meaning we give it an invalid user id

**Special Feature: Email notifications using the Nodemailer library/API.**

Where does the information come from?
- Users and a dedicated email that is used only for sending notifications to the users
- Information is sent, to users' emails, upon things in an event changing/being added/being deleted

What form should it take?
- Will have a notifications page for itself where users can select from a checkbox what they want to be sent to their email (email used will be their account email).

If the information is on the server, what will the client need to send to retrieve that data?

If the information is on the client, how will it be sent to the server?

Does all information need to be stored on the server?
- Information will be coming from a database where all other things are stored.

What processing needs to be done to make the data useful?
- Data used here will most likely be data that is already in other data tables (event finalised and event cancelled).
- Once that happens, we use the API to send out emails to the people participating