**Detailed Design**

**Preston Checkan, Bramdon Diaz, Ronny Elsheikh, Rickie Mobley**

**Project 1: Group Matching App**

**COP4331C, Fall, 2021**

**Contents of this Document**

Design Issues

Detailed Design Information

Trace of Requirements to Design

**Detailed Design Issues**

<In this section, provide more details to the design issues discussed in the high-level design. Document the results of each design prototype. Document design decisions and associated risks. Provide sufficient detail so the maintainer won't have to say "why did they do it this way?">

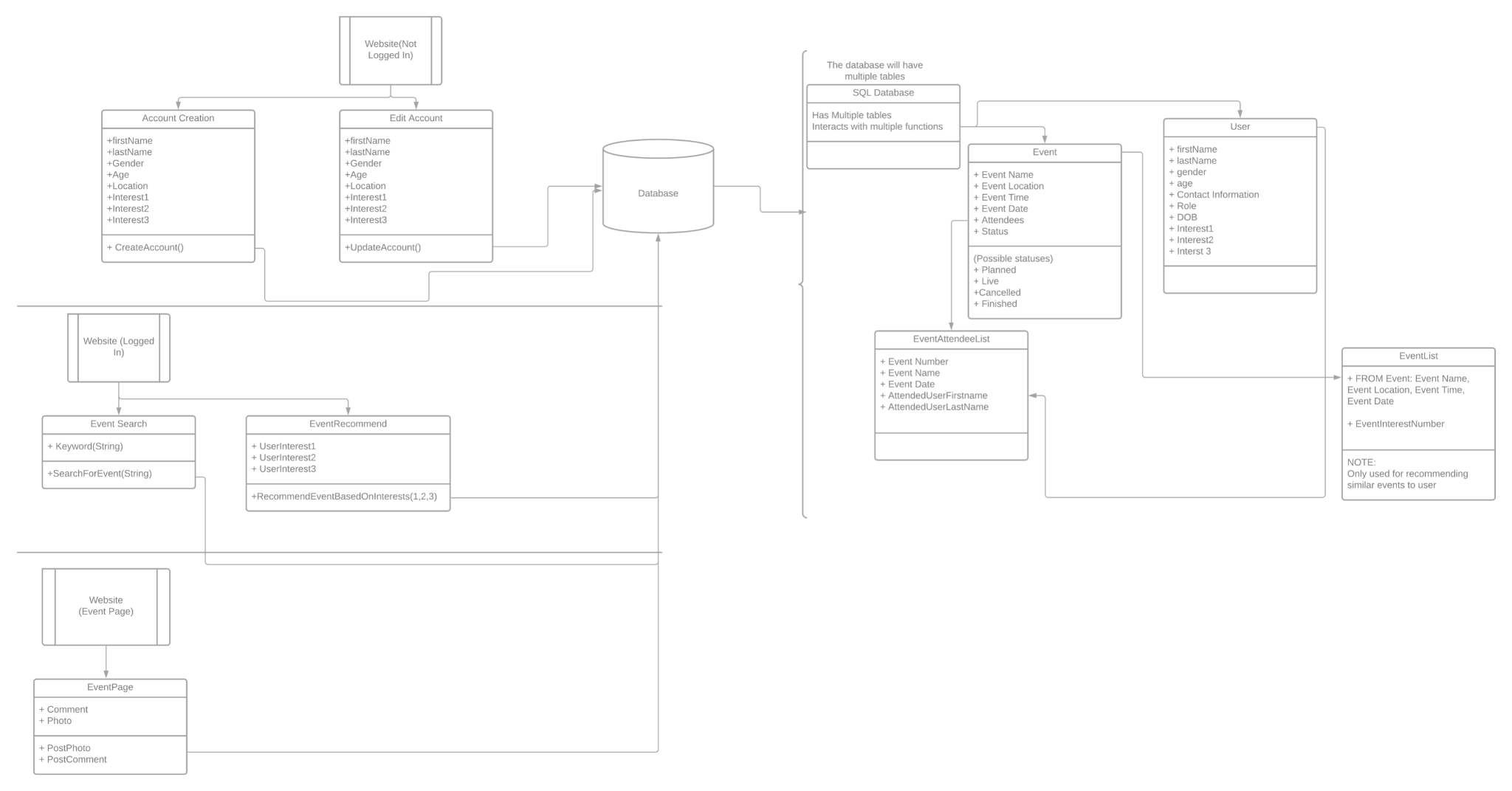
The way in which we designed the program was with the focus on maintainability, reliability, and portability. We are keeping most of our segments of the design unique in order to make it easier to maintain. Say a bug were to arise in the event search portion of the program. Since the other parts don’t rely directly on this segment we can correct the issue without having to rework other parts of the program. As far as reliability we choose 3 very well known languages to base the program on, that way there aren’t bugs that arise on the foundation of the program. We are also using a dedicated website hosting service to make the program always online. This also ties into the portability of the software, as there won’t be any portability issues with a device as long as it has internet access.

The pros to this type of design are described above, but lets go over some of the cons to this approach. For starters, this design causes it to be not as reusable when it comes to writing code for each section. This will in turn take more time to develop than if we made it more reusable. Another downside is since the majority of the design relies on the website and database being online, if either were to go offline the service would be unusable. Now that is a rare case but it is still an issue. Overall, We believe we took the appropriate steps to create a design that is efficient, reliable, maintainable, and portable. This will in turn create the best possible user experience.

**Detailed Design Information**

<In this section, include diagrams, listings, etc. to provide the complete details of the design of your product. A detailed class diagram is required. >

< The design documentation must be well organized, modular, and of sufficient detail such that the programmers will be able to start the implementation phase.>

  
Link to the detailed design diagram: <https://lucid.app/lucidchart/ad133d0e-240d-4746-9568-07fce3fb4847/edit?invitationId=inv_fdf8e50d-1311-4f0d-81fa-7fc40f63dc64>

**Trace of Requirements to Design**

<In this section, provide a trace of each requirement in your SRS to the design: for each requirement, indicate which module(s) will fulfill that requirement?>

Based on SRS document

* searchEngine(string)
  + Input: User search for event
* searchResult: list
  + Show recommended events
* editInformation()
  + Account log in
  + Account log out
  + Verify Correct Login
  + Invalid login, prompts user to re enter password
* dataBaseEditor(string, string, string, string, string, int, list)
  + The system will have a login system so that each person can access their account only. All the information will be private to them only, and if they forget login, or want to edit, then there would be a way to access the account through other methods, whether it be security question or password change.
* eventRecommender(list)
  + Input: User search for event
* recommendationResult: list
  + Event Recommended/Event Signed up for
* createAccount(string, string)
  + Account Created
* getFirstName(string)
  + Account Creation
* getLastName(string)
  + Account Creation
* getGender(string)
  + Account creation
* getAge(int)
  + Account creation
* getInterestOne(string)
  + Event Recommended/Event Signed up for
* getInterestTwo(string)
  + Event Recommended/Event Signed up for
* getInterestThree(string)
  + Event Recommended/Event Signed up for
* Event
  + Input: Event created
  + Event opened for others to sign up for
  + After event User comment/Photo upload
* EventAttendeeList
  + Verify User is attending event
* User
  + Event attended
  + After event User comment/Photo upload
  + Verify User is attending event
  + Event not currently attending, prompt user to attend event