# **Gigs n’ Chill**

# **Software Design**

## **CSCI-P465/565 (Software Engineering I)**

## **Project Team**

Deepan Elangovan

Harsha Valiveti

Jackie Drs

Andrew Gotts

**1. Introduction**

**This section introduces the design approach to the software system.**

### **1.1 System Description**

**Provide a description of the system, and the problem that is being solved by developing the system (may be copied from the Project Plan or Software Specification if those documents are accurate).**

This is an event finder page where people can either look for an event near them or post an event. The problem being solved by creating this is to get smaller events that have declined due to covid like movies, plays, and music out to the public.

### **1.2 Design Evolution**

**This section is intended to document the rationale behind the selected design solution.**

#### **1.2.1 Design Issues**

**Include in this section any issues/requirements/constraints that drive the design of the system. This may include desired operating environment, interface protocols, the need to be remotely accessible, etc.**  
 The design must have the following:

* Login in page
* Profile page
* Home page
* Search bar
* A way to reserve an event
* Post an event
* Bookmark an event
* Invite a friend to an event

#### **1.2.2 Candidate Design Solutions**

**Briefly (a paragraph or two) describe each of the possible design solutions that your team discussed/explored.**

We considered a number of development options, but in terms of the tech stack, we focused on Django (a Python framework) for the backend. We considered utilizing either base HTML/CSS/JavaScript or react for the front end. We are still experimenting with different hosting services to check what fits our needs, some of the candidates are AWS, Heroic and IU server.  
 For our team we talked about starting with a homepage that would lead to either being able to login or signup. This way people can access the site without an account first. Then this would lead to a main page, where a user can access their profile and search can events. Another idea would be after signing in to go to the person profile page.

#### **1.2.3 Design Solution Rationale**

Provide some reasoning for the design option that your development team selected. Tie in the selected approach's ability to meet the requirements, and most effectively address the design issues.  
 We ended up deciding after signing in to go to the homepage. This is due to the fact it will be better for customer usability. In most sites where a user must log in, they are directed to the main page afterward, so if we then went to the profile page first it would be confusing for users since that is not the way it is usually done.

### **1.3 Design Approach**

#### **1.3.1 Methods**

**Describe the methods that are employed in order to capture your design. What techniques did you use (prototypes, object-oriented design, design patterns, etc) and why?**

We used the common design pattern of having separate login and signup pages. In each of those we also have the standard design of top to bottom for information input. Aligning important areas in the web application when they reach, this way no time is wasted in searching the events. For design implementation, as we are using React, mostly the application would be Single Page Application, where the components are loaded whenever it is required. This brings best user experience. We have also started using Figma to map out what we want on each of the pages so we all can see before we deploy it.

#### **1.3.2 Standards**

**Describe any design standards to which your design adhered. These may include safety standards, operator interface standards, naming conventions, structure and hierarchy of components, etc.**

We applied to regular naming conventions, like login, sign up, home page, profile, and search. Also, we adhered to common hierarchy, for example the decision to go from logging in or signing up to going to a main page which is what most users will be used to.

#### **1.3.3 Tools**

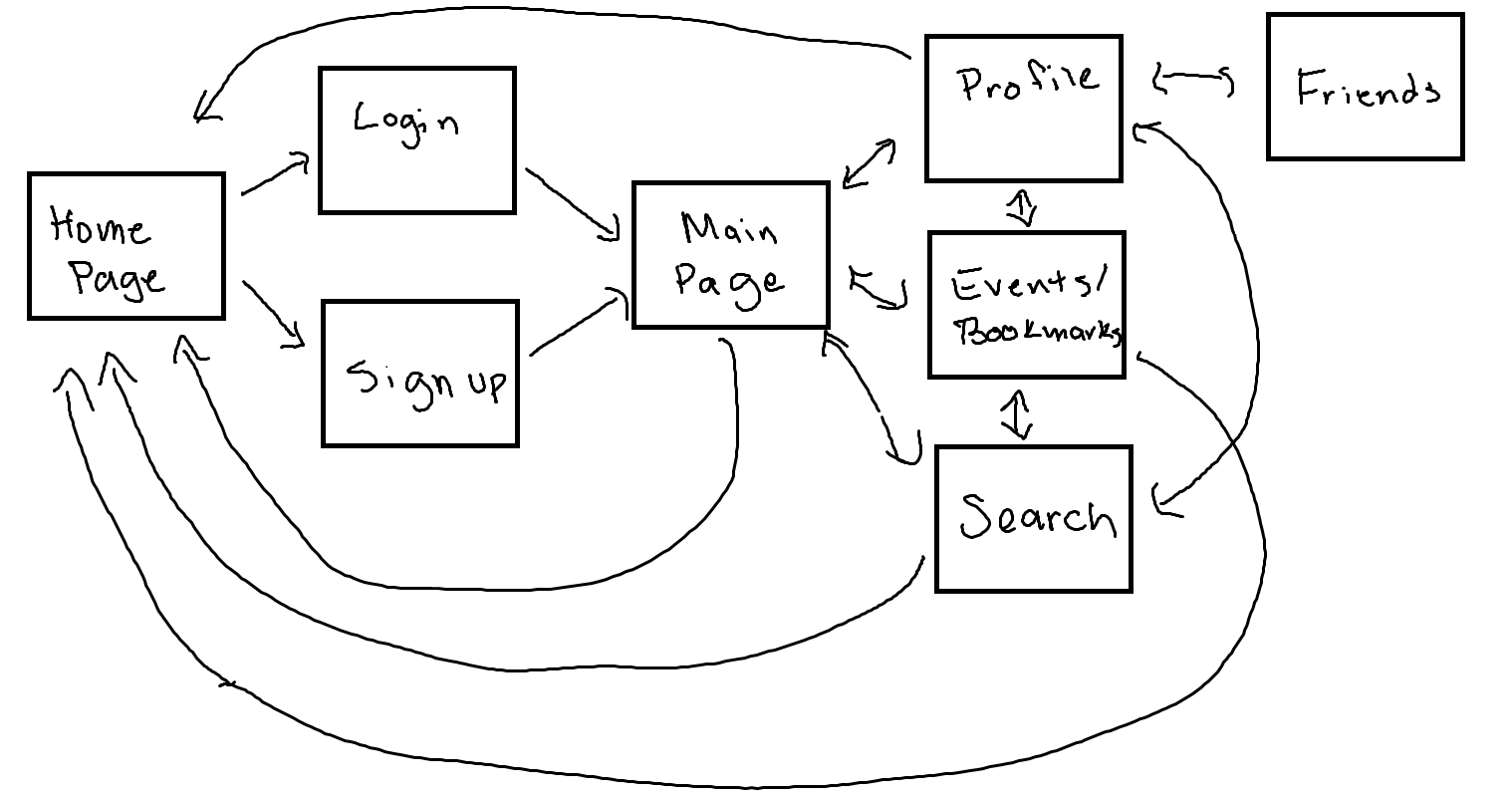
**Describe any tools that you plan to use to assist you in developing the design and specify exactly what products will be generated by the tools.**

We will be using Figma to map out the design before implementing it. We will be using a combination of React JS this will help us implement the flow of the pages. Along with css which will help us style the pages. For backend communication we are using Django, that will fetch the results for a user input. REST implementation for integrating backend and front-end, this is the most used approach in any web applications.

## **2. System Architecture**

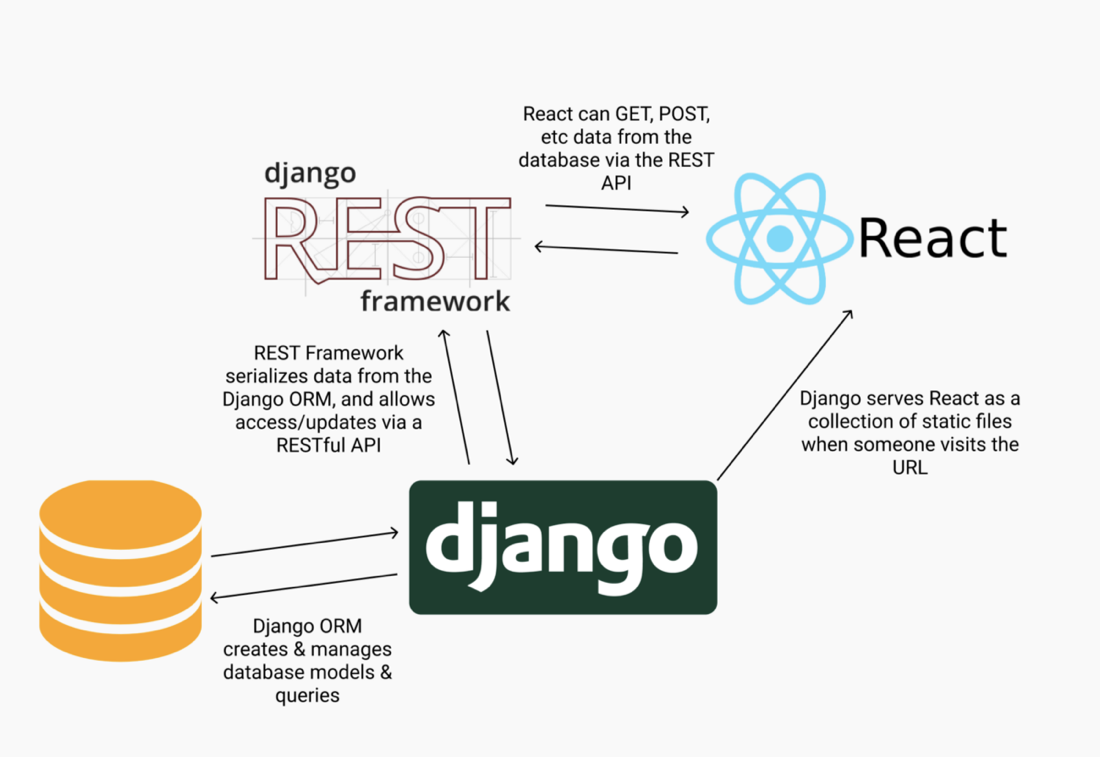
### **2.1 System Design**

**Provide a description of the high-level design of your system. Include a system diagram that captures the major components and the external interfaces. Describe the operation of the system (high-level) and the interaction between components.**

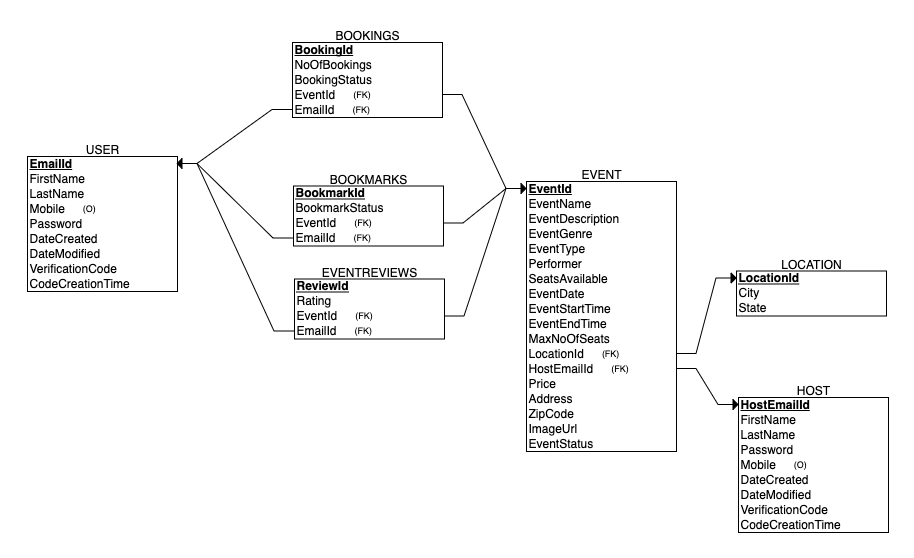


It starts at a basic homepage where you can only login or signup from there. Then after logging in or signing up you are taken to a homepage that can navigate to profile, events/bookmarks, or search through events. From each of those you can log out and go to the homepage once again. From the profile you can see your friends.

The diagram below illustrates the top-level design for our project.



The diagram below is the relation schema for database of the project.



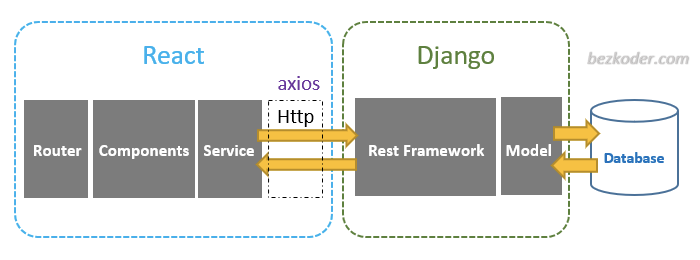
### **2.2 External Interfaces**

**Describe in detail all external interfaces that interact with your system. Be sure to include a description of the interface and a summary of its relationship with your system. Be sure to include specifics about the communication between your system and the external system including protocols, data structures exchanged, and timing/handshaking patterns.**

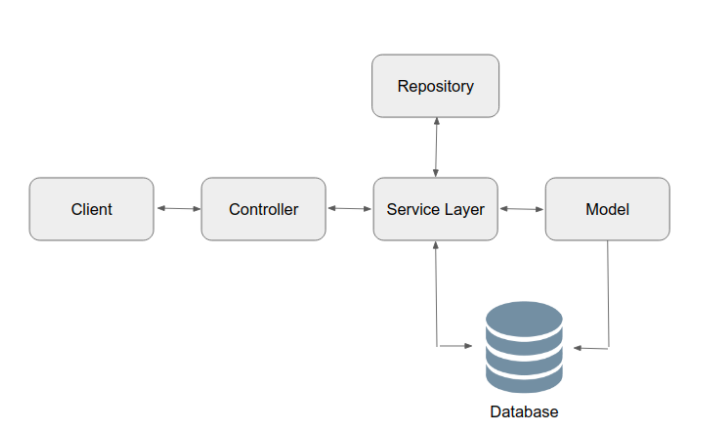
We do not intend to use many external APIs in this project in order to keep communication between services as simple as possible. Instead of creating an account through our website, we will use Google Login to allow our users to login with a verified user account. This will be accomplished by utilizing Google's API, which will evaluate the Google account and return a login token to the server, allowing users to access our website.

Social media – there will be an option to login through a previous social media account. Transferring over personal information and username

## **3. Component Design The Component Design section details the proposed design of each system component. A system component is a functional partition of the system. Components may be organized as you see fit - a component may be a collection of objects, or a single object. However, a system must be composed of multiple components (that is, a system cannot be one component). The layout of this section is at your discretion, but please include the following (at a minimum) information for each component:**



* **Component Name**  
  **Provide a unique component name**
  + Application Properties
  + Models
  + Exceptions
  + Payload
  + Repositories
  + Services
* **Component Description**  
  **Include a description of the function of the component. Feel free to include any diagrams/charts/tables that will provide understanding of the component you are developing. Make the description detailed enough so that the operational flow of the component is evident. If your component performs any kind of calculations, uses algorithms, or executes a mathematical model, be sure to document those methods here. Also, make clear whether or not the component is a stand-alone process or thread, or if its operation is included in the "main" execution loop.**  
  The application properties provide information for connecting to the database, such as the URL, login, and password. The controller gives the area to construct rest endpoints for your application, while the service layer handles all of the business logic. All database activities are handled by the repository (CRUD operations). The model is in charge of the objects in our application.
* **Responsible Development Team Member**  
  **Decide who on your development team (one person, please) is responsible for the design and implementation of this component.**  
  Andrew Gotts
* **Component Diagram**  
  **Graphically depict the design of the component in terms of interfaces with other components and external interfaces. Also, consider including a diagram depicting the internal operations and/or class relationships in the component.**



* **Component User Interface**  
  **If this component includes a user interface, include some details about the interface including what aspects of the component will be available through the interface, a description of each of the user screens that are expected for this component, and a description of each of the user notifications and/or messages that are planned for this component.**

Include some details about the user interface, such as what aspects of the component will be accessible through the interface, a description of each of the user screens that are expected for this component, and a description of each of the user notifications and/or messages that are expected for this component. Users that use the Google API or OAuth will be able to sign up and log in on the landing page.

* **Component Objects**  
  **Describe the objects/classes that comprise this component. Provide a listing of expected data members and methods for each class. Note in the description if a given object/method does any of the following (**Note: Some of this information may be captured in class diagrams or other parts of the design - simply be sure it is included somewhere):
  + Makes an operating system call (cite expected system calls to be made)
  + Makes a hardware-specific system call (cite calls to be made)
  + Creates/alters/deletes a file (cite file names)
  + Explicitly calls the method of another object (cite name of other object)
  + Is explicitly called by another object (cite name of other object)
  + Passes data to another object (cite data structure and name of other object)
  + Receives data from another object (cite data structure and name of other object)
  + Is derived from another object (cite name of other object)
* **Component Interfaces (internal and external)**  
  **Describe the component's interfaces with other components including a description of protocols to be used, methods to be called, and/or data structures to exchange. In addition, any operating system calls should be identified in this section. If the interface requires a particular data exchange sequence or timing pattern, include a sequence diagram to descibe it. Describe this component's relationship to the external interfaces also (if external interfaces were sufficiently covered in Section 2.2, then simply reference that section).**
* **Component Error Handling**  
  **Describe the steps taken in the design to incorporate fault tolerance, data corruption prevention, and incorrect operation avoidance. Please organize these into error cases.**  
  **Example:**  
  **Error Case 1: Input Validity Check: MyClass.initialize() method will range check its integer input such that 0 < x <= 5**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Team Member** | **Diag** | **User Interface** | **Objects** | **Interfaces** | **Error Handling** |
| Home Page | A landing page for when people enter the site | Andrew |  | Will have buttons to press based on whether the user wants to sign up or log in. |  |  |  |
| Login In | This is where the user will log in and it checks the database to see if a person exists | Harsha, Deepan, Andrew |  | Will have areas for the user to input data and submit the information to the database | Retrieves data from the database for a person | This will interact with the backside database and the frontside to give the person the data they require | Will output an error if the input does not match the one in the database |
| Sign Up | This is where the person will sign up to create a new user in the database and check to see if the is already a user that exists, as well as a proper password. | Harsha, Deepan, Andrew, Jackie |  | Will have areas for the user to input data and submit the information to the database | Creates a new user in the database | This will interact with the backside database and the frontside to give the person the data they require | Will output an error warning for an invalid password or a username/ email already in use. |
| Database of Users | Holds all the information of the users. | Deepan and Andrew |  | Not for the users/ no Ui | Contains the information of a user | It will connect with the front end to provide information on the users | Giving acknowledgement for success/failure of search/new entry. |
| Search for events | A user can search for events/type/name of venue/ Location/ Price | Deepan, Andrew and Harsha |  | A search bar for user to search | Displays events, venue for respective search | This will interact with the backside database and the frontside to give the person the data they require | Handling errors for invalid search |
| Profile | The page where users can go to see their information and their events | Andrew, Jackie |  | Can pick a date on a calendar and show events for that date | Retrieves events for the user | This will interact with the frontside and backside to provide the correct events for each person | Handling error for events not properly being shown |
| Homepage | The page where the users will be directed after logging in | Harsha |  | Can look through all of the events avaiable | Retrieves all the events that have been published | This will interact with the frontside and backside to provide the correct events that are available | Handling error for events not properly being shown |
| Database of Events | This will hold all of the events for the webpage | Deepan |  | No UI | Contains information for each of the events | It will connect to the front end to provide information on the events | Show if invalid data is entered |
| Create an Event | Allow the host to create and add an event | Harsha |  | Prompts for user input and details about the event | Collects information needed to add an event | Will take the information given and push it to the event database | Show if there is data missing or not proper data |
| Register for an Event | Allow a user to register to the events | Harsha |  | Prompts user for the amount of tickets and to register | Shows the user details for the event and availability | Will book an event for the user | Indicate ticket can’t be booked if space is not available |
| Event Management for Host | Allow host to manage the Event and See the detail, cancelling events | Harsha |  | Displaying Event Details and Feature to cancel the event | Details of the event, ability to see attendees | Cancel the event | Checks whether the event is active |
| Event Reservation | Allows user to Cancel the reservation and confirmation for booking | Harsha |  | Displaying the Event Details along with the booking details  And cancelling reservation | Details of the Event and Booking details | Cancel and Booking details | Matching the booking tthe registered User |
| Inviting Friends | Allows user/host to invite people to the event | Harsha |  | Prompts modal to inviting the people | User/host object to invite | Inviting through Email | Sending mail to the friend and mail checking |
| Chat | Allowing User/Host to chat one-on-one | Harsha |  | Provides the ability to communicate each other | Checking for the registered users/host | Interface to search and initiate chat | Communicating with the respective user |
|  |  |  |  |  |  |  |  |

## **Revision History**

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Change Description** |
| Initial Set Up | 2/20/22 | Set Up the document after doing the login page and things we are going to do in the future |
| Added Profile and Homepage | 3/6/22 | Added the profile and homepage as they were completed in this sprint |
| Added figma to design plan, Event Database, Create and Register Event | 3/27/22 | Added database for events and a way to create and register for the events. Also added a new way to design which is figma |
| Storing Event images in Firebase as a supplement to actual DB | 3/27/22 | Storing Images in firebase and storing URL to access in MySQL |
| Using Third party Chat package to implement | 4/10/22 | Using third party DB and API’s to communicate through |
| Frontend Hosting with Firebase | 4/10/22 | Deployed Frontend application with Firebase |
|  |  |  |
|  |  |  |
|  |  |  |

**Page Author/Creator:** [Adeel Bhutta](http://homes.soic.indiana.edu/aabhutta/)Last Modified: 8/23/2016