1. Project name:

RSVP Collection System (RCS)

2. Team members:

Benjamin Bruyns
Peggy Lewis
Spenser Morey

3. Abstract:

When events are held, it can be challenging to determine the exact number of people who will attend them. So, invitations are sent out to invite guests and to determine how many people will be in attendance. However, if information is required in the RSVP, compiling information from the invitees is tedious.

To solve this problem, we are creating the RCS program. This program will be used to create events with required RSVP information. The program will then be able to receive RSVPs from invitees. The information will then be stored, and the event host can compile information from the RSVPs.

4. Description:

The RSVP Collection System (RCS) is an event management program that can accommodate various events, from small social gatherings to campus-wide events. Its primary function is to create and distribute invitations, track RSVPs, and manage the logistics of an event. This tool simplifies planning, freeing hosts to focus on more complex responsibilities, and provides party-goers with a unique experience. Despite its simple design, this tool incorporates often-overlooked planning details, aiming to create a seamless experience every time.

The RCS will assist event hosts by creating events through a user-friendly interface. Hosts can enter an event name and an optional note for gift ideas, instructions, or other details. They can select a start date and time and, optionally, an end date and time for the event. The host can enter the full event address to specify the event location. The system will automatically validate the address, checking for and correcting any typos or formatting issues to ensure the address is accurate. Hosts can provide a descriptive location name, such as “McGregor Park & Riverwalk,” “The Morgan University Center Ballroom,” or “Smith Family Residence,” offering guests context about the event's physical setting.

To set the event's visual theme, the host can select from a library of pre-generated images provided by the RCS. Alternatively, the host can upload their pre-designed custom image as a theme. The system offers additional customization settings for the event invitation, including:

- Allowing guests to indicate the number of children they will be bringing
- Limiting the number of additional guests permitted to attend (from 0 to 5)
- Enabling guests to RSVP, register for the system, or share the event with others

The host can actively manage event notifications, enabling real-time push or email notifications whenever guests RSVP. Hosts can communicate with guests through an event chat for any last-minute seating updates, changes of plans, or time conflicts, for example. The host can also send personalized text or email reminders to the entire guest list or select individuals. The system offers capabilities for guest management where hosts can manually add or remove attendees, adjust the number of additional party members for each invite, and update guest information as needed.

Hosts can include additional links, such as gift registries or other important event information. For events requiring interactive coordination, like potlucks, hosts can create customizable sign-up sheets. Guests can then select items from these lists, which are dynamically marked as claimed or removed, preventing duplication.

Upon completion of each step, the system will generate an invitation based on the host's inputs and selected theme. The host can preview the invitation, save it for later, or immediately send it to guests via text message, email, or a printed QR code that directs recipients to the event webpage.

This comprehensive event management system will streamline creating and sharing events, providing hosts with a centralized platform to handle all the necessary details. From intimate gatherings to large-scale functions, hosts can create events with customizable invitations, track RSVPs, manage an event's guest list, or even create interactive sign-up sheets for potlucks or gift registries. With its user-friendly interface, automated address validation, theme customization, and flexible communication tools, the system simplifies event planning while enhancing the experience for both hosts and guests. On every occasion, it ensures that event management is both efficient and enjoyable for everyone involved.

5. Feature list:

Features completed by the end of the semester:

- Website for Host and Recipients to view and manage the upcoming event(s)
- Hosts can create and distribute invitations digitally
- Hosts can print invitations with a quick response (QR) code linking to the website for RSVP management and recipient information
- RSVP form to gather essential information from Recipients, including but not limited to

    - Name
    - Contact information
    - Number of people in the party (if Host allows)
- Verify the address of the event
- Recipients can save the invitation

Features completed if time allows:

- Chat for each event
- Map to the location (Ability to open in Google Maps/Waze)
- Allows photos from Host for invitations
- Upload images from the event
- Add direct links to invitations for online stores or external gift registries
- Waiting list if all spots are full for the event

Features we would like to implement in the future:

- Share the event (if enabled) on social media (FB/Instagram/Twitter etc.)
- Create a sign-up sheet for invitees' items to bring to events
- Notifications:
    - Host: Receive notifications about recipients' acceptance/rejection
    - Recipients: Receive notifications about the event or any updates from the host
- FAQ for additional information for guests during the Rsvping process
- Automated “Thank You” cards for attendees
- Track guests' dietary preferences and food restrictions

6. Technology:

Platform: web-based. We chose web-based for its scalability and wide range of use regardless of operating system.

The framework for server-side development will be the Python framework Django. This allows us to easily program the server-side function using Python for rapid and simplified development.

The IDE we will be using is Pycharm. This will allow us to easily write the Django code and will allow us to host the website locally to test features while developing.

The languages we will use will be HTML, CSS, JavaScript, and Python.

Consider NodeJS as an alternative.

7. Server information:

- Server-side framework: Django

- Server hosting service: Google Firebase. ”Non-option”

8. Data sources:

- Address Verification: Google Maps API
- Theme Image Sources: Pexels, Unsplash, Pixabay, and Canva

9. Team members’ backgrounds:

Benjamin’s Background:

Experienced with data manipulation and other backend responsibilities. I have some experience integrating databases to the front end but none with servers. I am sure this will be learnable. I have minimal experience with markup languages, but I am sure I can become familiar with them without much issue. Tentatively, my primary responsibilities will be data storage and retrieval. I will also be involved in integrating data storage and the front end.

Peggy’s Background:

Experienced in working with Java and Python programming languages using NetBeans and IntelliJ IDE for developing independent, stand-alone applications. However, I have no experience building projects using HTML, CSS, or JavaScript, nor integrating any database server. However, my primary responsibility will be creating the front-end user interface, researching the best tools for the project, and implementing functionality using APIs and libraries. This project will be a new learning experience in every aspect of the development process.

Spenser’s Background:

Experienced in web development, backend programming, and database integration. I have worked on several Django projects and deployed one to a hosting platform. I have minimal experience with HTML, CSS, and JavaScript, but I am pretty familiar with Python, the Pycharm IDE, and the Django framework. My primary responsibility will be to program the server-side functionality and assist with database integration and storage.

10. Dependencies, limitations, and risks:

Dependencies:

- Google Maps API: Needed for address validation; any issues could impact accuracy.
- Image Sources (Pexels, Unsplash, Pixabay, Canva): Rely on external platforms and licensing for event themes; limits customization.

- Google Firebase: Hosting depends on Firebase, which could affect uptime and performance. “Non-option”
- Django Framework: Django updates might affect stability. The team might use only one version of Django to prevent this.
- Third-Party Libraries: Compatibility issues could arise with libraries.

Limitations:
- API Integration Challenges: Integrating Google Maps and image services may be tricky.
- Firebase Storage Limits: Firebase's free tier might limit data capacity. “Non-option”
- Customization Options: Limited by available images and tools.
- Notification Delays: External services may cause delays in real-time alerts.

Risks:
- Dependency on APIs: Outages or changes in APIs could disrupt functionality.
- Data Security: Risks of breaches from storing personal info.
- Time Constraints on Features: Limited time might force cutting features.
- Cross-Browser Compatibility: Different browsers might cause inconsistent behavior. We will pick a browser.
- Limited Front-End Experience: Learning curve with HTML, CSS, and JavaScript.

11. Timeline:
Week 1:

- Benjamin: Set up the backend environment and research Django.
- Peggy: Research front-end tools and set up basic UI structure.
- Spenser: Set up Django on Firebase, and test server functionality.

Week 2:

- Benjamin: Begin database setup for events/RSVPs.
- Peggy: Design event creation UI.
- Spenser: Implement event creation and RSVP handling.

Week 3:

- Benjamin: Integrate Google Maps API.
- Peggy: Build theme customization UI.
- Spenser: Implement quick response (QR) code for printed invitations.

Week 4:

- Benjamin: Implement RSVP tracking.
- Peggy: Create UI for guest list management.
- Spenser: Backend logic for RSVP updates and guest management.

Week 5:

- Benjamin: Set up notifications for RSVP changes inside the webpage.
- Peggy: Add notification settings in UI.

- Spenser: Build email/push notification system.

Week 6:

- Benjamin: Add advanced features (sign-up sheets, registries).
- Peggy: Finalize UI and additional customizations.
- Spenser: Implement event chat, maps, and guest controls.

Week 7:

- All: Test and debug full system integration.

Week 8:

- All: Finalize features and fix bugs.

Week 9:

- All: Full testing and deployment to Firebase.

Week 10:

- All: Documentation and presentation prep.

The remaining 2 weeks are set aside for complications.