Note: The final document will need to be formatted as specified here: <http://aaronbloomfield.github.io/slp/slides/spring/02-deliverables.html#/stsreport>

Specific formatting: <http://www.acm.org/sigs/publications/proceedings-templates> (Tighter alternate style)

* Abstract
* Introduction
  + Innisfree is a lifesharing community for adults with disabilities {innisfree}
    - 4 “coworkers” live with 2 volunteers and all contribute to tasks for the community
      * Cooking, gardening, etc
  + Has ~40 residents that they take care of, as well as ~40 volunteers / staff
  + Current solution: Use a pen and paper system of scheduling
    - Too slow
    - Hard to keep records
    - Secretarial load
    - Hard to generate statistics and reports
    - Risk of miscommunication
  + SLP course structure
    - Group work, meeting structure
    - 2 week iterations, TDD (specific type of development methodology we are employing – in early slides)
* Background
  + Primarily used as a scheduling system to manage resident appointments
    - Provides easily accessible information and reminders to volunteers / staff
    - Manages checking out cars to drive residents to these appointments
  + Ruby on Rails {rails}
  + Testing done via Factories
* Related work
  + Google / Apple / other mainstream scheduling online scheduling options
    - Too generic, doesn’t allow for management of residents, doctors, and other involved parties
    - No user privilege levels
    - Optimized to be managed by one user; can’t have multiple users with their own individually generated calendars
    - Hard to filter calendars
  + Outlook
    - Too complex for requirements
    - Volunteers don’t want to spend time navigating a complex desktop-based system, they need a quick method to schedule and check appointments while on the go
    - No way of supporting car-reservation systems
    - No authentication-based privileges
* System Design
  + MVC model
  + Login system
    - Three levels of authentication
      * Admin / Staff
        + Create users, houses, residents, and doctors
        + Generate reports -- more information below
        + Medical coordinator is a field in the user table...they get extra e-mails!
      * Volunteers
        + Create appointments
        + Modify appointments
        + Many permissions are limited to the house that the volunteer is in
      * Workstation heads
        + Read-only access
    - Every account has an entry in Users
      * Authentication level is a field in the Users table
  + Users
    - Admin create and delete users
      * Create a password that the user themselves may change
    - Users can see other users (but cannot modify without admin priveleges)
    - Users can modify their own information, including their password.  Their password can also be modified by admin, but cannot be seen by admin
  + Appointments schedule (core functionality)
    - * Appointments must have a resident, doctor, date, time, and type
      * Appointments are displayed on a calendar on the landing page
        + Used the fullcalendar Rails gem to display {fullcalendar}
      * Creating appointment
        + Simple, mobile-friendly form
      * Canceling appointments
        + Field in the schema so these appointments persist
        + Displayed with strikethrough
  + Houses
    - Residents and volunteers are assigned to a specific house
    - Volunteers can perform CRUD operations on appointments and residents within their house
    - Houses created by admin (staff)
  + Doctors
    - List of doctors with basic information (name, type, contact)
    - Appointments specify a doctor that the resident is visiting
    - Only admin and create/delete doctors
  + Cars
    - Calendar to show when certain cars are available
    - Can add cars available for use by volunteers / staff
    - Volunteers and staff can make reservations to use the car for an appointment
      * The same car cannot be reserved twice for the same time slot
      * Anyone can modify another reservation in case of extenuating circumstances such as traffic
  + Reports
    - Dynamic report generation
    - Shows all the appointments with the given specifications
      * Can filter by houses, resident, doctor, type, and date
  + Email notifications
    - Morning digest of that day’s e-mails
    - Reminder to schedule a follow-up
    - Notification when appointment scheduled in a house
    - Friday e-mail with the next week’s appointments sent to medical coordinator
  + Mobile view
  + Do we want a full schema diagram or discussion?
  + Add any particularly interesting design decisions (will need to discuss in meeting)
* Procedure
  + Staff and volunteers will use the website on a daily basis to:
    - Make appointments
    - View their current appointments
    - Make changes to residents, houses, etc.
    - Reserve cars
  + Data on residents, houses, etc. is updated as it changes
  + Volunteers sign up for email alerts to remind them of their residents’ appointments
  + Staff can generate reports of appointments for data backup / hard copies
* Results
  + Three months to reach basic development goals
  + Three months of essentially an alpha test where use and major development occurred concurrently
  + 6 weeks of “transition”/beta testing in which the customer used the system extensively with only minor development and bug fixes
  + The customer can much more efficiently manage their resident’s appointment schedules
  + The volunteers and staff can now access the management system from anywhere using their phone / computer
  + Requires actual numbers from user testing
* Challenges/barriers (we can rename this section...I added it)
  + Data access
    - Tedious and time consuming
  + Access to site for volunteers
    - One computer in each house
    - Only about half the volunteers have smartphones
* Conclusions
  + Designed an appointment scheduling system to allow Innisfree to manage their residents’ appointments via a web interface, which allows them to save time going through the process of scheduling appointments via the previous pen and paper method
  + Summarize major points from technical report
  + Summarize requirements, and the extent to which we satisfied them
  + *This should be written after the paper is written and not now*
* Future work
  + Creating a dedicated mobile app
  + Making the calendar more descriptive of appointments
  + Add and talk about more ideas for more features and improvements that could be added to the system in the future
  + Volunteer shift calendar
* Acknowledgements
  + Ruby / Rails gems
  + Monika, Emily, Wes, and Eric at Innisfree Village
* References
  + "Innisfree Village." *Innisfree Village*. Web. 10 Apr. 2015. <<http://www.innisfreevillage.org/>>. {innisfree}
  + Ruby On Rails (Version 4.1) [Computer Software]. (2014). Retrieved from <https://gorails.com/setup/ubuntu/14.04>. {rails}
  + FullCalendar (Version 2.3) [Computer Software] (2014). Retrieved from <http://fullcalendar.io/>. {fullcalendar}
  + Sites for gems we used
  + Tutorials we used
* Author list:
  + “Bethany Connor  
    University of Virginia  
    [bac5rc@virginia.edu](mailto:bac5rc@virginia.edu)”
  + “Robert Emerson  
    University of Virginia  
    [roe2pj@virginia.edu](mailto:roe2pj@virginia.edu)”
  + “Will Emmanuel  
    University of Virginia  
    [wre9fz@virginia.edu](mailto:xvp2he@virginia.edu)”
  + “Xavier Palathingal  
    University of Virginia  
    [xvp2he@virginia.edu](mailto:xvp2he@virginia.edu)”
  + “Domenic Puzio  
    University of Virginia  
    [dvp5qd@virginia.edu](mailto:xvp2he@virginia.edu)”
  + “Huiqing Zhang  
    University of Virginia  
    [hz9cx@virginia.edu](mailto:xvp2he@virginia.edu)”