SRS: 401 Senior Project - USChedule

Owners: *Annie Flora, Amelia Jay, Liam Namba, Cristalina Nguyen, Sophia Prochnow, Christian Santander*

Contacts

Product Management Owner	FUZZIES
Engineering Owner	FUZZIES
Stakeholders	Kelly Jones, MD kelly.jones2@med.usc.edu; Dr. Andrew Forney aforney1@lmu.edu; Dr. BJ Johnson

5.1 Introduction

Description

A concise summary of the feature/project, capturing why we are doing this, what we are doing and the benefits for consumers and other stakeholders.

LAC+USC psychiatry residency program has an inpatient locked facility at Augustus Hawkins Hospital. During the course of the year, 24 residents (interns and second-year residents) rotate through the facility under Chief Resident Dr. Kelly Jones. Dr. Jones is responsible for the creation of a call schedule for the year which takes into account the following responsibilities:

- Day float (person on call during the day when the rest of the team members leave)
- Night float (person on call who covers the entire hospital overnight)
- Long call and short call (people who cover the hospital over the weekend)
- Each resident rotates over the course of a two month block
- Vacation block requests (try to take preferences into account if possible)
- Balance each ward with second and first year residents so that each ward is equivalent in terms of resident knowledge/experience

The creation of this schedule by hand requires lots of time and resources from Dr. Jones, and she needs a system that can accelerate this process. The output of this PRD is to create such a system.

- **Problem:** Presently, the chief formulates the schedule by trial and error (plugging in different roles into the excel spreadsheet until a permutation that works occurs). Can take hours!
- **Goal:** Create a webapp that can take the residents' preferences and then generate a call schedule by the constraints on the previous slide
- **Hopes**: Dr. Jones would like to solicit vacation requests and try the best to honor them if someone requests a certain week that can work in the schedule
- **Publication potential**: hoping to submit to a medical education journal and/or conference!

Presentation proposal

Are There Past or Future Phases of this Feature?

If this feature is a success, what's next? If this is a follow on, what was the previous phase? Future phases should be covered in new PRDs or in this PRD, but please briefly describe what (a) the first phase of this feature and (b) any subsequent phases.

Phase 1 - MVP: Fulfill minimum scheduling, user and constraint requirements
Later Phases:

- More constraint capabilities
- Better UX
- Allow residents to request/swap shifts with another resident without bothering the admin
- Scale to work for any type of company
    - Add people to groups (ex: Sales Associates, Managers, etc.)
    - Put constraints on the groups instead of per individual
- Monetize (?)

5.2 CSCI Component Breakdown

CSCI USChedule is composed of the following CSCs:

5.2.1 GUI Client CSC - **Node.js/React on AWS S3**

5.2.1.1 Login Page
5.2.1.2 Home Page
5.2.1.3 Scheduling Page
5.2.1.4 Resident Management Page
5.2.1.5 Output Page

5.2.2 Server CSC - **Python/Flask on AWS Lambdas**

5.2.2.1 Get data from database 'GET'
5.2.2.2 Put data into database 'POST'
5.2.2.3 Algorithm to determine optimized schedule
5.2.2.4 Validation check for schedule

5.2.3 Database CSC - **MySQL on AWS RDS**

5.2.3.1 Schedule schema
5.2.3.1.1 Employee table
5.2.3.1.1.1 ID
5.2.3.1.1.2 First Name
5.2.3.1.1.3 Last Name

5.2.3.1.1.4 Year
5.2.3.1.1.5 Office ID
5.2.3.1.1.6 Shift ID
5.2.3.1.2 Organization table
5.2.3.1.2.1 ID
5.2.3.1.2.2 Name
5.2.3.1.2.3 Address
5.2.3.1.2.4 City
5.2.3.1.2.5 State
5.2.3.1.3 Shift table
5.2.3.1.3.1 ID
5.2.3.1.3.2 Type
5.2.3.1.3.3 Length
5.2.3.1.3.4 Duration

5.3 Functional Requirements by CSC

Functional Requirements for Admin Users:

User Story: Kelly Jones is the Chief Resident of Inpatient Psychiatry at LAC+USC Medical Center. Kelly is supervisor to 24 residents who maintain a rotating schedule of working hours in each ward of the hospital. She must create the schedule and uses a system that given some constraints, requests, and list of residents, outputs a schedule that fulfills all scheduling requirements that Kelly wants.

5.3.1 The GUI shall display a login page
5.3.1.1 The login page shall display an input box for username
5.3.1.2 The login page shall display an input box for password
5.3.1.3 The login page shall verify that the username and password are correct and existing
5.3.1.4 The login page shall display a forget password option
5.3.1.5 The login page shall display a button that will be used to login and continue to the next page
5.3.1.6 The login page shall display a button that allows the user to create an account
5.3.1.7 The login page shall be clear, intuitive and visually appealing
5.3.2 The GUI shall display a home page
5.3.2.1 The home page shall display a button to “View Current Schedule”
5.3.2.2 The home page shall display a button to “Create New Schedule”
5.3.2.3 The home page shall display a button to “View All Residents”
5.3.2.4 The home page shall display a button to return to the login page
5.3.2.5 The home page shall be clear, intuitive, and visually appealing
5.3.3 The GUI shall display a scheduling page
5.3.3.1 The scheduling page shall display a list of the current residents that will be scheduled
5.3.3.2 The scheduling page shall display options for each resident

5.3.3.3 The scheduling page shall display a “Ward Preference” dropdown
5.3.3.4 The scheduling page shall display a “Float Preference” dropdown
5.3.3.5 The scheduling page shall display an input box for “Vacation Preference”
5.3.3.6 The scheduling page shall display a button that will be used to continue to the next page
5.3.3.7 The scheduling page shall be clear, intuitive, and visually appealing
5.3.4 The GUI shall display a resident management page
5.3.4.1 The resident management page shall display a list of all the current residents on schedule
5.3.4.2 The resident management page shall display a button to “Add New Resident” to the system
5.3.4.3 The resident management page shall display an input form to input criteria for said new resident
5.3.4.4 The resident management page shall display a button to “Remove a Resident” from the system
5.3.4.5 The resident management page shall display a “Confirm” popup and button to remove said resident
5.3.4.6 The resident management page shall display a button to “Edit Resident Profile” in the system
5.3.4.7 The resident management page shall display the current profile with ability to change information
5.3.4.8 The resident management page shall display buttons that will be used to continue to the next page
5.3.4.9 The resident management page shall display filters to apply on the list of residents displayed
5.3.4.10 The resident management page shall display buttons that will be used to navigate around the page
5.3.4.11 The resident management page shall be clear, intuitive, and visually appealing
5.3.5 The GUI shall display an output page
5.3.5.1 The output page shall display the schedule that was just generated
5.3.5.2 The output page shall display a button for user to return to the scheduling page
5.3.5.3 The output page shall display a button for user to “Download” the schedule
5.3.5.4 The output page shall display a button for user to “Go to Home Page”
5.3.5.5 The output page shall be clear, intuitive, and visually appealing
5.3.6 The server shall get data from and push data to the database using API calls
5.3.6.1 The API shall support a ‘GET’ call to return information on residents
5.3.6.2 The API shall support a ‘POST’ call to add new residents to the database
5.3.6.3 The API shall support a ‘PUT’ or ‘PATCH’ call to update resident information
5.3.7 The server shall support our algorithm for creating schedules to run on it
5.3.7.1 The algorithm shall implement constraint propagation
5.3.7.2 The algorithm shall support making changes to the existing schedule and shall check if constraints are still being met
    This will be called a validation check
5.3.8 The server shall support unit testing

5.3.8.1 The unit tests shall be comprehensive and ensure that the algorithm is performing optimally
5.3.9 The database shall have a schema named ‘Schedule’
5.3.9.1 The Schedule schema shall contain a table named ‘Employee’
5.3.9.1.1 The Employee table shall contain a column named ‘ID’
5.3.9.1.2 The Employee table shall contain a column named ‘First Name’
5.3.9.1.3 The Employee table shall contain a column named ‘Last Name’
5.3.9.1.4 The Employee table shall contain a column named ‘Year’
5.3.9.1.5 The Employee table shall contain a column named ‘Office ID’
5.3.9.1.6 The Employee table shall contain a column named ‘Shift ID’
5.3.9.2 The Schedule schema shall contain a table named ‘Organization’
5.3.9.2.1 The Organization table shall contain a column named ‘ID’
5.3.9.2.2 The Organization table shall contain a column named ‘Name’
5.3.9.2.3 The Organization table shall contain a column named ‘Address’
5.3.9.2.4 The Organization table shall contain a column named ‘City’
5.3.9.2.5 The Organization table shall contain a column named ‘State’
5.3.9.3 The Schedule schema will contain a table named ‘Shift’
5.3.9.3.1 The Shift table shall contain a column named ‘ID’
5.3.9.3.2 The Shift table shall contain a column named ‘Type’
5.3.9.3.3 The Shift table shall contain a column named ‘Length’
5.3.9.3.4 The Shift table shall contain a column named ‘Duration’

NOTES FROM DR. JONES:

- Must be in 6 month blocks: July - December and January - June
- Must display each block as a week
- Must only be accessible by the admin
- Must schedule residents by year experience: 1st or 2nd year
- Must support 24 residents, with 7-8 residents working at any given time
- Must support adding and deleting residents from the database: Residents stay in system for 2 years
- Must ensure that every week has someone accounted for the day float, night float, long call, etc
- Should try its best to prioritize vacation preferences
- Should not have residents repeat ward assignments (night float two times, etc)
- Must assign second years to the adolescent ward (ward F) for one of the two months
- Must have at least 2 people at each ward at one time
- Should place a first year and second year at each ward
- Must place residents at the same ward 2 months at a time
- Should try to cater to their requests (one might prefer ward D). this isn't imperative
- Must give residents one week off, in any two month block
- Must be dynamic and allow admin to make revisions after creating schedule

WANTS

- Clear policy to take vacation requests before time
    - Maybe a form that residents fill out, Kelly inputs the actual information
- Survey monkey to see how residents like the scheduling

- Clear policy for residents to swap shifts: not on the official schedule

5.4 Performance Requirements by CSC

5.4.1 The user shall be logged in within 10 seconds
5.4.2 The user shall be directed from the home page to their desired page within 5 seconds
5.4.3 The user shall be able to navigate between pages in 4 seconds
5.4.4 The system will read information from the database within 30 seconds
5.4.5 The database shall make changes submitted by the user within 10 seconds
5.4.6 The schedule shall be generated within 1 minute
5.4.7 The schedule shall be displayed for viewing within 15 seconds
5.4.8 The schedule will be re-generated for validation check within 1 minute
5.4.9 The schedule shall be downloaded to the users device within 1 minute

5.5 Project Environment Requirements

5.5.1 Development Environment Requirements

Category	**Requirement**
Front End	NodeJS, React, S3 Cloudfront
Back End	Python, Flask, Zappa
Server	AWS Relational Database Services

5.5.2 Execution Environment Requirements

Category	**Requirement**
Processor	Pentium processor at 90 MHz or higher
Hard Drive Space	10mb
RAM	16mb
Display	800x600 256 Colors

The Following are the software requirements for the Scheduling System

Category	**Requirement**
Operating System	None

Graphics	DirectX 3.0 or Higher
Compiler	None

Miscellaneous

Compliance and Non-Functional Requirements

List any compliance or non-functional requirements (e.g., accounting, legal or contractual conditions, marketing or sales considerations, etc.) that support the core use cases. This section provides and business level context for those requirements (e.g., an external partner requires a specific SLA or sales through this product have specific tax implications for consumers and for merchants). The numbering of these requirements is contiguous with the User Stories.

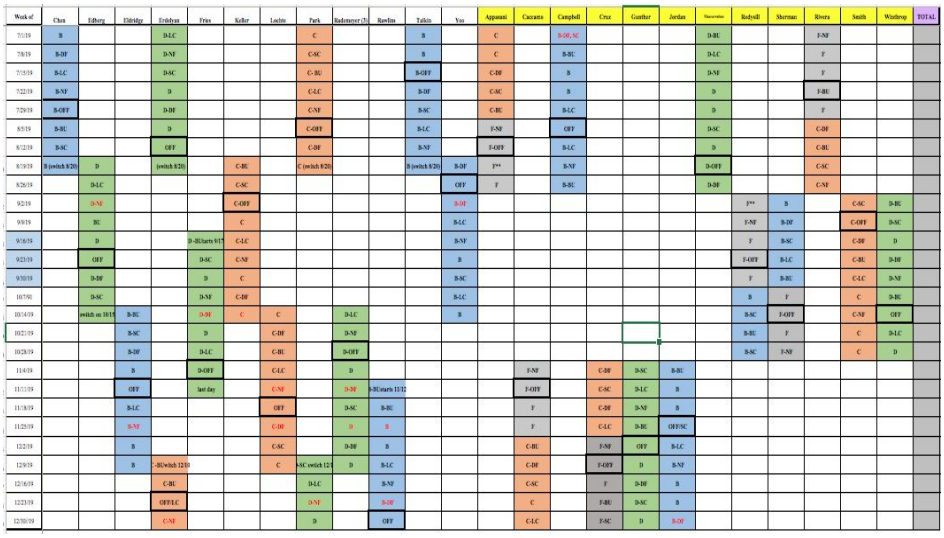
1. HIPPA / PII

UX Requirements

List any specific UX or design requirements. The numbering of these requirements is contiguous.

1. [MUST] Input areas must be clear and legible to provide residents and admins a smooth experience

Designs or Wireframes



Reference & Supporting Documentation

Include any reference material that will help provide context or other background. This might include strategy docs or related PRDs.

Presentation proposal