UNIVERSITY OF St.Thomas

Symp+Rack Final Project Report

By
Ahmed Algogandi, Michael S. Schindeldecker, Sarah Seitz, Uma Elangovan

Under the guidance of **Professor Chi-Lung Chiang**
Course Name: **SEIS 610-02 Software Engineering**

Document Change Control:

Author	Document section	Link
Entire Project Team (Collaborative effort)	Introduction Tool Evaluation Lesson Learnt Conclusion	Introduction Tool Evaluation Lesson Learnt Conclusion
Ahmed Algogandi	Product System Design / Installation / Product Prototype	Symp+Rack System Design and Installation
Michael S. Schindeldecker	Product Process Flow Diagram Product Data Flow Diagram	Symp+Rack Product Flow Diagram Symp+Rack Data Flow Diagram
Sarah Seitz	Product Introduction Product Mobile Prototype Product Usability Testing	Introduction to Symp+Rack Symp+Rack Mobile Prototype Overview Symp+Rack Product Usability Testing Overview
Uma Elangovan	Product Architecture Project Management Overview	Symp+Rack Component Architecture Overview Project Management Overview

Table of Contents

1. Introduction (All)

This section outlines the project background, scope, timeline, risks and dependency if any.

1.1 Project Background

SymTrack aims to provide the patient journey from the day 1 of their chronic illness until recovery by both the Physicians and the patients. Tracking the information daily allows both patients and doctors to obtain a clear picture of the both the severity and frequency of symptoms over an extended period. This app stores the data in the backend to generate weekly, monthly, and biannual reports in a line graph and sent as email to the physicians.

1.2 Project Scope

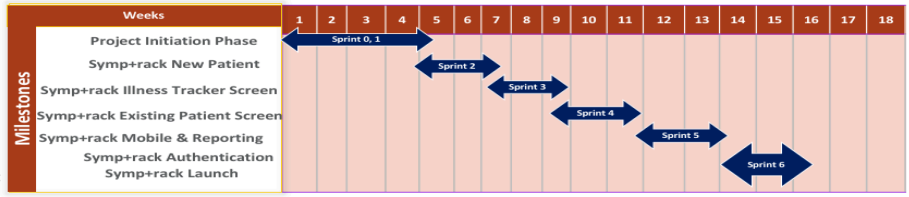
- Requirement Gathering and Story creation
- System Design, Build, Test and Deploy. DevOps and Source code management
- Design front end UX, service layer, backend data store for reports and analytical operation. Support both Web based as well as Mobile based browsing
- Wireframe creation for front end visuals that demonstrate Mockup of emails complete with graphs sent to patients and doctors. Mockup of set up and data entry
- Data security and management (HIPAA compliance applicable to product)
- User Acceptance Testing
- Application Launch and Monitoring

1.3 Project Requirements

- System should be able to track up to 5 symptoms entered by the patient.
- System should store the current treatment plan.
- System should allow the patient to rate the severity of the symptom on a daily basis till the date of their next appointment. Includes additional features to accept manual entries for blood sugar levels, blood pressure and pulse from the patient.
- System should generate weekly, monthly, and biannual reports in a line graph.
- Reports should be available to the patient and a copy should be sent to the medical team via email.

1.4 Project Timeline

Our project team adapted agile methodology that engaged a team of 4 individuals for 6 sprints with 2 weeks per sprint – model implementing symp+rack through scrum delivery framework.



1.5 Project Risks and Dependency

- Being HIPAA compliant app, the security measures and data security required complete auditing and certification process
- Dependent with actual Physician and Patient data sets to validate the functionality of the symp+rack app

2 Symp+Rack Product Manual

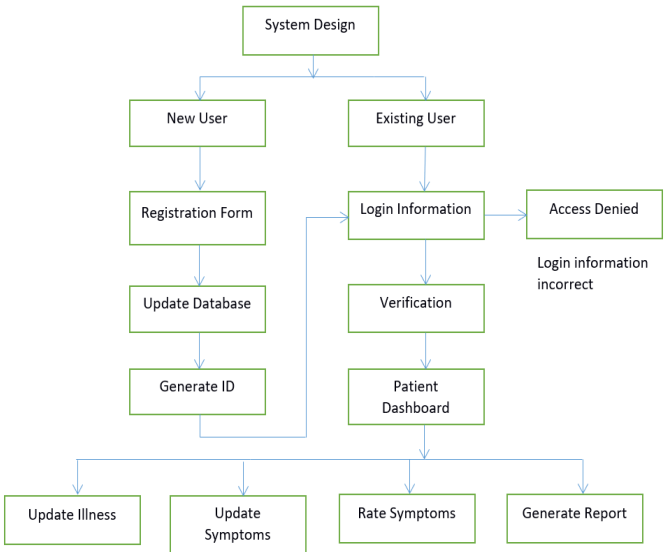
This section outlines the detailed description of the symp+rack product and its system design, functionality, architecture and deployment model.

2.1 Symp+Rack Product Overview (Sarah)

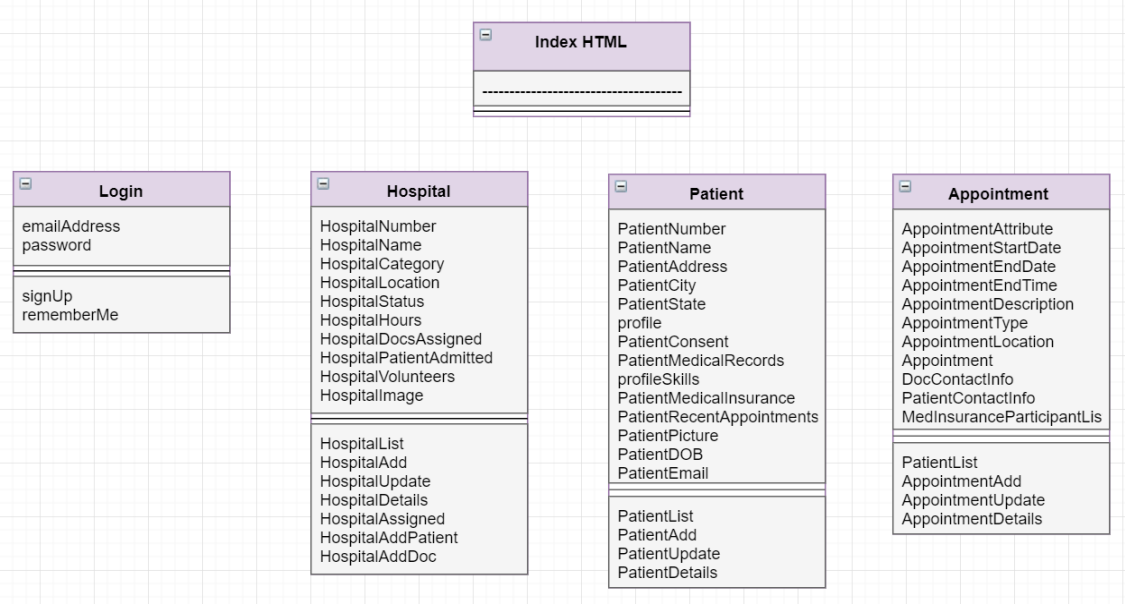
Millions of Americans face the day to day suffering that stems from one or more chronic illnesses. They struggle to make sense of their symptoms. They fight for care standards, work with care teams, and try to balance their illness or illnesses with their lives. Symp+rack wants to help make their lives a little easier while having an impact on the level of care received by providing an easy to use symptom diary available on either the web or a mobile device. Symp+rack will collect user data about symptoms, store it, and display the data collected in an easy to use and understand graph. Users will be able to track their data on a weekly, monthly, and annual basis. Additionally, Symp+rack will push the collected data to doctors, nurses, and other care staff allowing accurate information sharing between patient and care team regarding the severity and frequency of symptoms. Having this information will cut down on the amount of time spent checking in with patients and will offer more concrete data points for treatment plans.

2.2 Symp+Rack Product Flow Diagram (Michael)

Below flow chart represents the data /process flow diagram within the Symp+rack application.



2.3 Symp+Rack Data Flow Diagram (Michael)

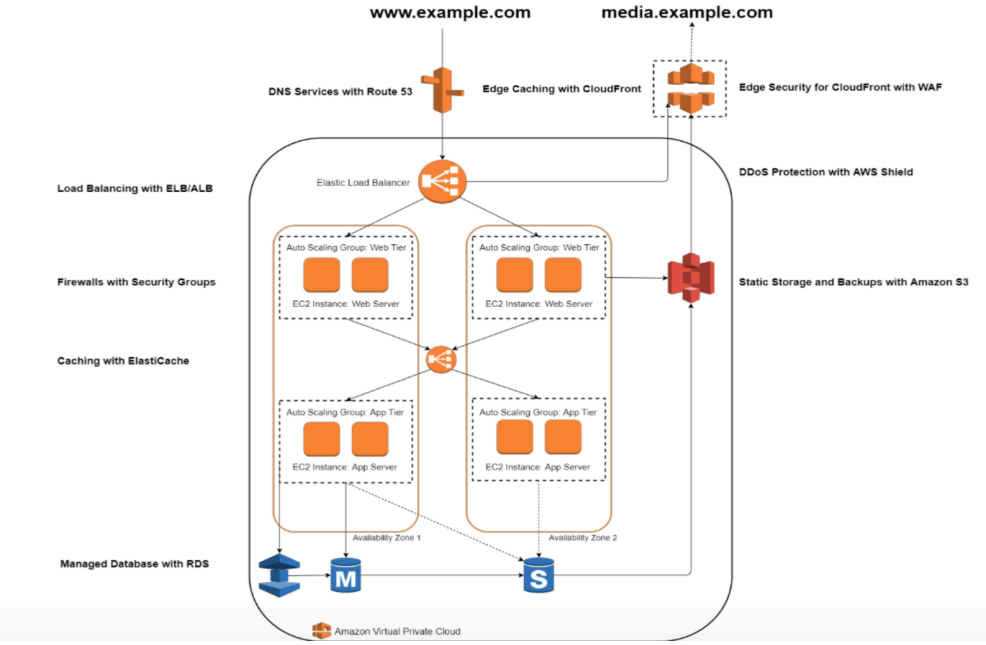


2.4 Symp+Rack Component Architecture Overview (Uma)

This product requires highly scalable, available infrastructure to store massive volume of patient day to day data in a real time fashion to the end users. Hence leverages AWS public cloud platform on a highly secured infrastructure

Below is the architectural component used to host the symp+rack application

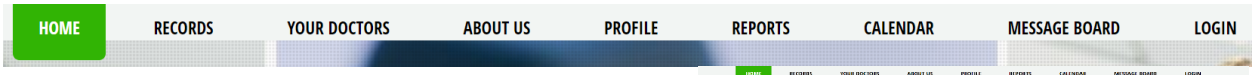
1. Load Balancing with Elastic Load Balancing (ELB)/Application Load Balancer (ALB) – Allows you to spread load across multiple Availability Zones and Amazon EC2 Auto Scaling groups for redundancy and decoupling of services.
2. Firewalls with Security Groups –Moves security to the instance to provide a stateful, host-level firewall for both web and application servers.
3. Caching with Amazon ElastiCache – Provides caching services with Redis or Memcached to remove load from the app and database, and lower latency for frequent requests.
4. Managed Database with Amazon RDS – Creates a highly available, Multi-AZ database architecture with six possible DB engines.
5. DNS Services with Amazon Route 53 – Provides DNS services to simplify domain management.
6. Edge Caching with Amazon CloudFront – Edge caches high- volume content to decrease the latency to customers.
7. Edge Security for Amazon CloudFront with AWS WAF – Filters malicious traffic, including XSS and SQL injection via customer-defined rules.
8. DDoS Protection with AWS Shield – Safeguards your infrastructure against the most common network and transport layer DDoS attacks automatically.
9. Static Storage and Backups with Amazon S3 – Enables simple HTTP-based object storage for backups and static assets like images and video.



2.5 Symp+Rack System Design and Installation (Ahmed)

System Menu

SympTrack is page-based website/application, which consists of 9 pages. A Home page, three pages show relevant information to the users, a page, Reports, represents a form for collecting users' data, and one page shows the users' calendar. The second last tab has a communication platform, while the last handles the login area.

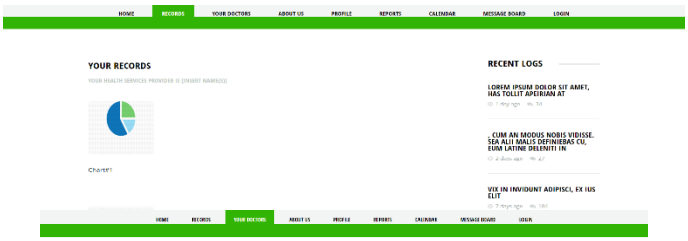


Home Page

The Home page consists of a header, containing a horizontal navigation bar, a body displays the application slogan, and a footer with a sitemap and contact information.

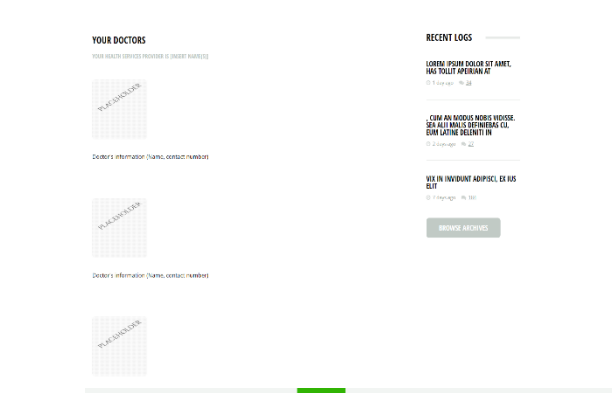
Records Page

The Records Page where the users can see their past medical records, symptoms, medication progress and browse their logs.



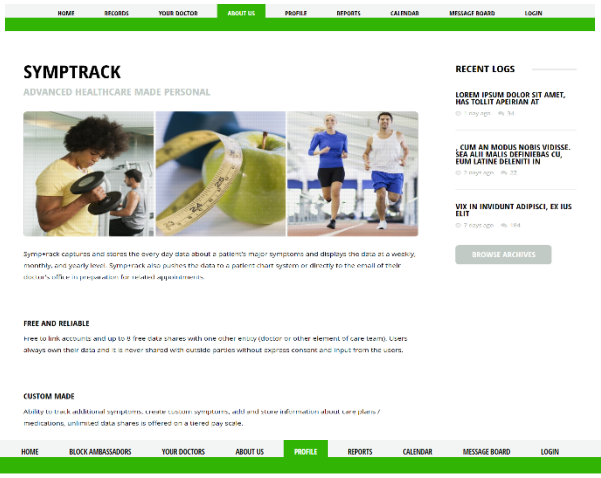
Your Doctors Page

The Your Doctors page display the users' health service providers' information



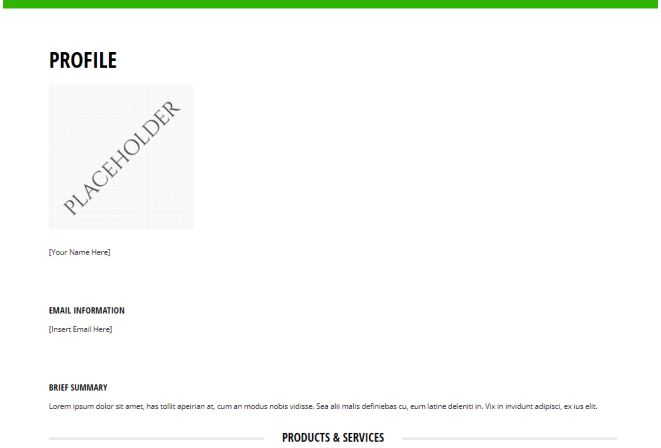
About us Page

The About us page gives customers more insight into who is involved with SympTrack and exactly what it does, expressed through short articles, accompanied by photographs.



Profile Page

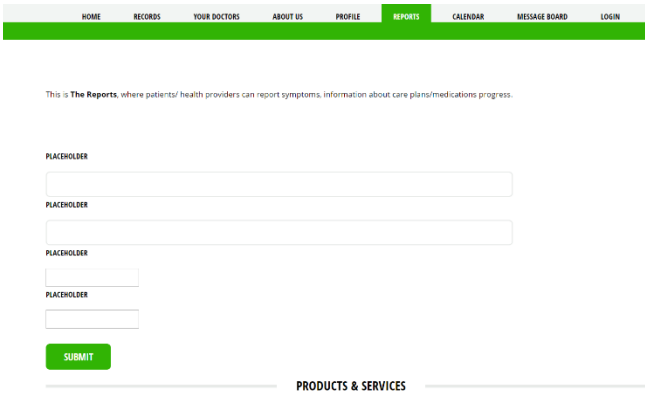
The Profile page entering the user's personal information as well as changing some application settings, such as source of contact preference information, and contact time.



Project Name: **Symp+Rack Project**
Prepared By: **Ahmed, Michael, Sarah, Uma**

Date Documented: **May 10th 2018**
Course Name: **SEIS 610-02 Software Engineering**

Reports Page

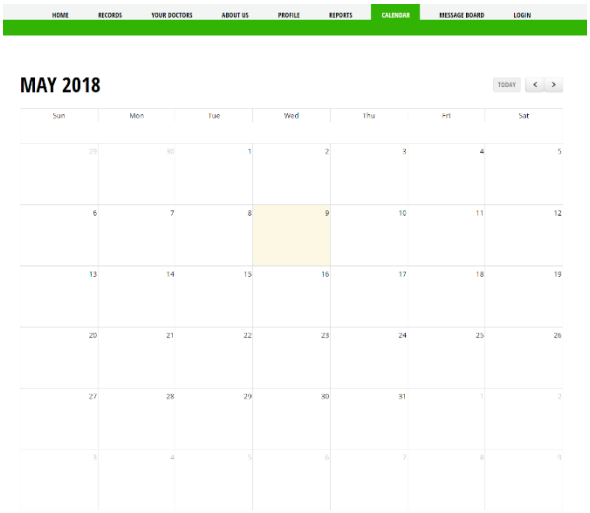


The Reports where patients/ health providers can report symptoms, information about care plans/medications progress.

Report Procedures

The report described in previous section will be accessed by typing http://www.symptrack.com/reports/ in address bar of the Internet browser. Each user can see only information about own health records. Data collected using the application are saved as in the database and can only be accessed by health services providers who are authorized by the user.

Calendar Page



The calendar interface helps the user to track their appointments and medication process. They can scroll through different months and select an entire date just by clicking the box. The flyout calendar widget is the same width as the input field so it doesn't take up that much space.

Message Board

For this product, we will be using a Message Board Hosting service. There are many message board hosting services available. Some will charge a fee, others will host the board free. Using a Message Board Hosting service, eliminates the need and expertise to purchase, install, and maintain the software, and there is much less need for technical expertise using a hosting service. We will be using a semi-threaded board, since the Board is "Discussion Topic" oriented, and allows replies to message topics.

Logging In

We are using a third-party framework of python for website development. The most useful part of this framework is that for account registrations and login, we can use a package known as registration-redux. All what we need to do is to install that package on the virtual environment of the PC with all the source code and include it in my project to do my work.

Changing User ID and Password

User ID and password can be changed only by contacting producer

2.6 Symp+Rack Mobile Prototype Overview (Sarah)

Creating an Account

Before setting up their profile, users will need to create an account with Symp+rack by providing a valid email address and creating a secure password. Once the email has been verified and the terms and conditions agreed to, users will be able to create a profile that will be associated with the account. The account will own the data gathered by Symp+rack and at no time with any of the information be shared any parties outside of the allowed care team and any other 3rd party entities as designated by the user. Permissions may be revoked at any time by the account holder.

Setting Up a Profile

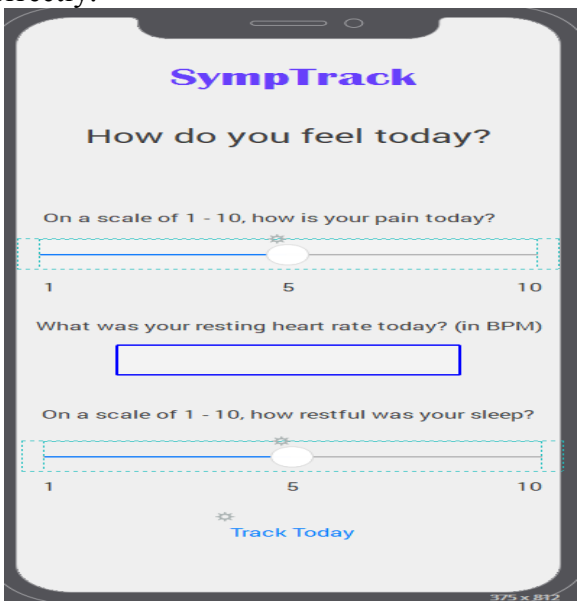
After the email address has been verified, users will begin the process of setting up their profile. Depending on their payment plan, they will be able to select up to 3 chronic illnesses from a predetermined list or create their own. The chronic illnesses selected will then generated a list of most common symptoms related to the illness. If the symptoms are not present, users will be able to create their own. Users will be limited to the number of symptoms they can track by the payment options they selected. For a free use, users will be able to track 3, with the potential to track up to 5.

Setting Up a Care Team

Setting up a care team will require some extra steps on the part of the user. They will need to coordinate with their care team the best way for the care team to receive this information. Symp+rack will offer the data collections in an email / PDF view and in a XML format. It is up to the patient to obtain and enter the email address used by their care team to manage incoming patient files. This page will include the functionality of linking an account or patient number to the email for any sorting software used by the care team.

Entering Symptoms on Mobile

Below is an example of the Symptom tracking screen on a mobile device. Questions for the symptoms will vary but all will have some sort of metric. Symp+rack will offer the ability to rate a symptom on a scale of 1 to 10, enter in a hard data point (such as a blood pressure or heart rate), as well as other symptom assessors. Users will also have the ability to go in an manually adjust their symptoms for the day in the event that symptoms change or the data was inputted incorrectly.



Graphical Display Page

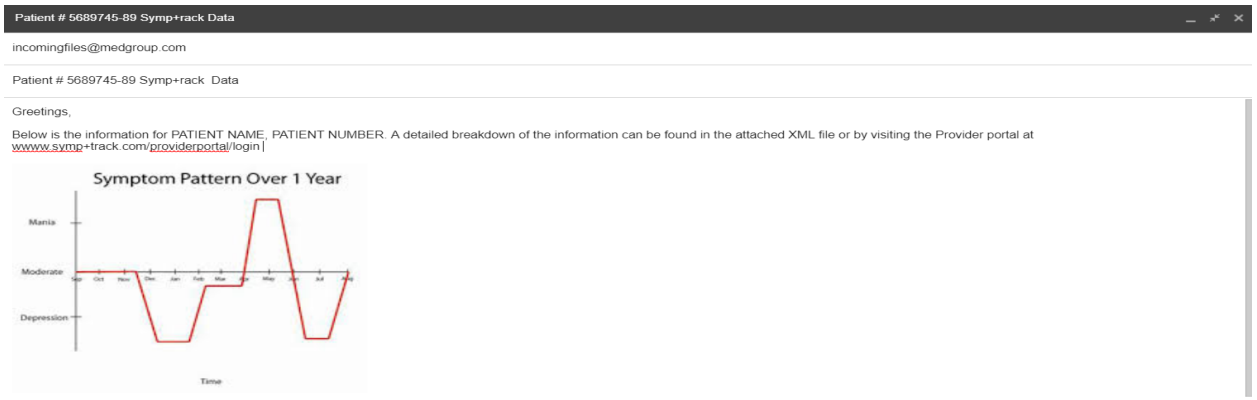
After selecting the "Track Today" button, a graph will update with the new data. The graph will initially display a graph for each symptom for the week. Buttons will allow the user to toggle between a month view and a year to date view.

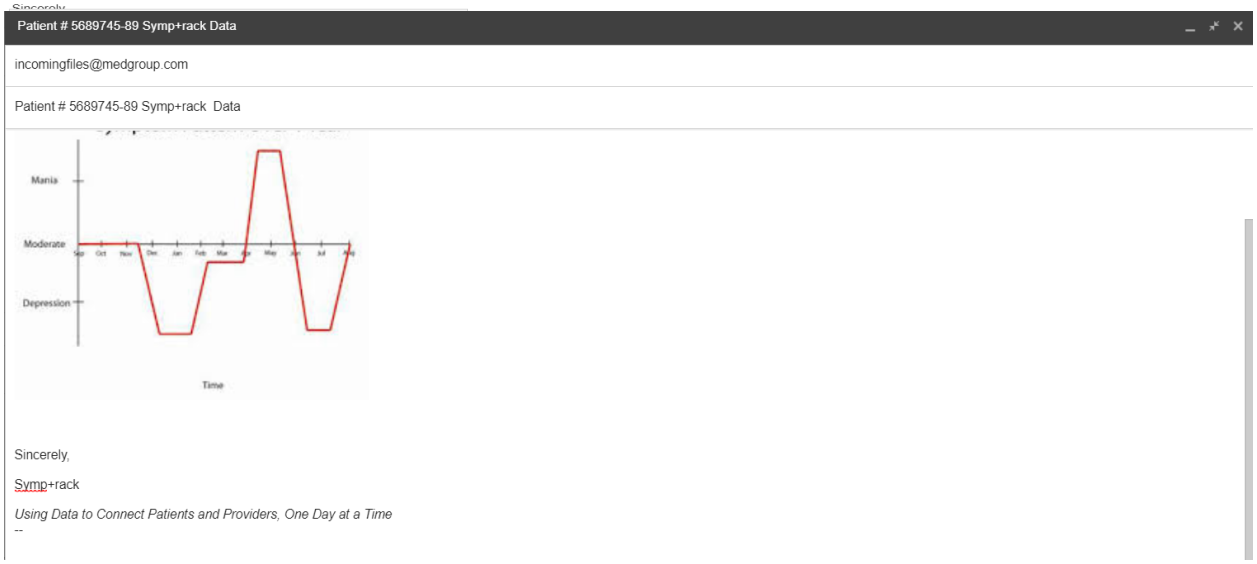
Reports Page

After the user has input the information needed for their designated destination, they can send reports out to the doctor's office. Users will have the ability to send out reports on a monthly, biannually, and yearly basis. They will also be able to schedule reports to be sent out by certain dates, like if they have reoccurring appointments. Users will also be able to print the reports from a weekly, monthly, and yearly basis in the event that a doctor's office does not accept electronic records.

Email

Below is an example of an email sent out to a doctor's office with the image of a graph in the email.





2.7 Symp+Rack Product Usability Testing Overview (Sarah)

To fully test the product, we will need to be able to get the experience stories from both the patients and the care teams. Doctor offices are frighteningly low tech throughout much of the country. There is a pretty big resistance to change, evidenced by the continued and steady

reliance on fax machines by many offices and hospitals. A deeper immersion into the way the patient and the doctors would use the product is required. User stories and use case diagrams have the greatest chances for success for in depth requirements analysis. The best way to do this would be to partner with research studies on chronic illnesses or on new drugs that are undergoing human trials. Symp+rack would offer the use of the product to monitor participants in the study in exchange for survey feedback regarding the use of the application. In addition, we could layer a heat map over the doctor and patient portal to measure what pages are most frequently used. Capturing the key clicks and actions would also prove beneficial as to the ease of use for both providers and patients.

3 Project Management Overview (Uma)

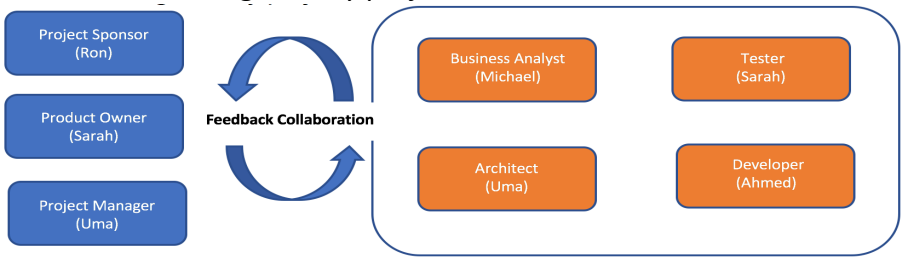
3.1 Detailed Project Plan & Work log

Project Charter/Assessment Sprint	Project Task	Start Date	End Date	Primary Owner	Status
Sprint 0 – Project Approval / Initiation Phase	**Requirement Analysis & Scope Approval**	**2-Feb**	**8-Feb**	**Sarah, Michael,**	**Completed**
	Deliver High Level Solution Architecture	9-Feb	15-Feb	Uma, Ahmed	Completed
	Project Estimation and Schedule Signoff	16-Feb	21-Feb	Sarah, Michael, Uma, Ahmed	Completed
	Environment Readiness	16-Feb	21-Feb	Sarah, Michael, Uma, Ahmed	Completed
Sprint 1 Epic (2 weeks Work Effort)	**New Patient Registration Login User story**	**19-Feb**	**2-Mar**	**Sarah, Michael, Uma, Ahmed**	**Completed**
	Wireframe - Log in Screen, Patient registration screen	19-Feb	21-Feb	Michael	Completed
	Develop Login and Registration UI	21-Feb	25-Feb	Ahmed	Completed
	Data Model, Backend Stored proc to generate Patient ID,	21-Feb	27-Feb	Uma	Completed
	Connectivity Testing, Master Test Plan, Test Strategy	21-Feb	2-Mar	Sarah	Completed
	Sprint Demo, Retro, Backlog	2-Mar	2-Mar	Sarah, Michael, Uma, Ahmed	Completed
Sprint 2 Epic (2 weeks Work Effort)	**New Patient Illness Tracker Screen User story**	**5-Mar**	**16-Mar**	**Sarah, Michael, Uma, Ahmed**	**Completed**
	Wireframe - Illness entry screen, Rating Screen, Symptom capturing Screen, Treatment Screen	5-Mar	7-Mar	Michael	Completed
	UI Development	7-Mar	11-Mar	Ahmed	Completed
	Backend development	10-Mar`	13-Mar	Uma	Completed
	Testing - QA Environment, Prod Test	12-Mar	15-Mar	Sarah	Completed
	Sprint Demo, Retro, Back log	16-Mar	16-Mar	Sarah, Michael, Uma, Ahmed	Completed
Sprint 3 Epic (2 weeks Work Effort)	**Existing Patient Login and Illness Tracking Screen User Story**	**19-Mar**	**30-Mar**	**Sarah, Michael, Uma, Ahmed**	**Completed**
	Wireframe for Existing Patient Login, Illness Track, Treatment Plan	19-Mar	21-Mar	Michael	Completed
	UI Development	20-Mar	26-Mar	Ahmed	Completed
	Backend development	21-Mar	26-Mar	Uma	Completed
	Mock Up data, Admin UI set up, QA Environment, Prod Test	24-Mar	29-Mar	Sarah	Completed
	Sprint Demo, Retro, Back log	30-Mar	30-Mar	Sarah, Michael, Uma, Ahmed	Completed
Sprint 4 Epic (2 weeks Work Effort)	**Mobile Services and Report Generation User Story**	**2-Apr**	**13-Apr**	**Sarah, Michael, Uma, Ahmed**	**Completed**
	Report UI	2-Apr	7-Apr	Michael	Completed
	Report Backend development	2-Apr	7-Apr	Ahmed	Completed
	Mobile Integration	2-Apr	7-Apr	Uma	Completed
	Mock Up data, Admin UI set up , QA	5-Apr	12-Apr	Sarah	Completed
	Sprint Demo, Retro, Back log	13-Apr	13-Apr	Sarah, Michael, Uma, Ahmed	Completed
Sprint 5 Epic (2 weeks Work Effort)	**User Authentication User Story**	**16-Apr**	**27-Apr**	**Sarah, Michael, Uma, Ahmed**	**Completed**
	Active Directory Integration	16-Apr	21-Apr	Michael	Completed
	User session management	18-Apr	24-Apr	Ahmed	Completed
	User, Role management	20-Apr	24-Apr	Uma	Completed

Project Charter/Assessment Sprint	Project Task	Start Date	End Date	Primary Owner	Status
	Security Testing	20-Apr	26-Apr	Sarah	Completed
	Sprint Demo, Retro, Back log	27-Apr	27-Apr	Sarah, Michael, Uma, Ahmed	Completed
Sprint 6 Epic (2 weeks Work Effort)	**Launch SymTrack Application**	**30-Apr**	**13-May**	**Sarah, Michael, Uma, Ahmed**	**Completed**
	Smoke, Performance Testing	30-Apr	2-May	Michael	Completed
	Bug Fix and Code Freeze	30-Apr	5-May	Ahmed	Completed
	UAT Sign off	7-May	7-May	Uma	Completed
	Release Plan	9-May	11-May	Sarah	Completed
	Deployment	13-May	13-May	Sarah, Michael, Uma, Ahmed	Completed
	Production Monitoring	13-May	13-May	Sarah, Michael, Uma, Ahmed	Completed

3.2 Project Team Structure

We are a team of 5 including our key project stakeholder



3.3 Team Roles and Responsibilities

Role	Role Description	Responsibilities
Product owner	Top level executive	Decides project strategic direction
Project Manager	Planning, budgeting, oversight and document all aspects of implementation	Co-ordinate, status reports, daily stand ups, Retrospective meetings, backlog tracking. Ensure sprint delivery
Business Analyst	Works with Business and IT team	Requirement Gathering, Analysis, UAT assistance
Architect	Solution Architects	Overall technical guidance and advisory
Developer	Engineer with coding skill sets	Design, development of product. Ensures unit testing is done thoroughly
Tester	Quality assurance analyst	Certifies the product functionally and technically. System, Regression and UAT testing is delivered without any defects

3.4 Project Testing Outcome

Team went through System testing followed by regression in order to make sure the functionality is thoroughly tested during each sprint. UAT was performed along with Business team.

TESTING SCOPE AND EXECUTION

- Validate the Symp+rack app delivers the IT needs
- Validate Infrastructure set up
- Validate Security testing
- System Testing → Regression Testing → UAT

System Testing

Entry Criteria:

A. Working Code with all major errors fixed

B. Environment provisioned for testing

Exit Criteria:

A. Complete individual unit testing for each components

B. Bug fixes are fixed and revalidated

C. All system test scenarios are executed

Regression Testing

Entry Criteria:

A. Complete System Testing

B. Web Api , Database env, Cloud provisioning for regression testing is available

Exit Criteria:

A. Validate all high and critical functionalities and certify the minimum viable product feature is achieved

B. Complete the data validation , security etc.

C. All major bug fixed and re validated

D. Complete Web / App connectivity for end user validation

User Acceptance Testing

Entry Criteria:

A. All regression testing scenerios should be complete

B. Stable UAT environment available

C. Production like data are ready to be send to UAT

D. All connectivity include network is highly available and validated

Exit Criteria:

A. All UAT test scenarios are complete

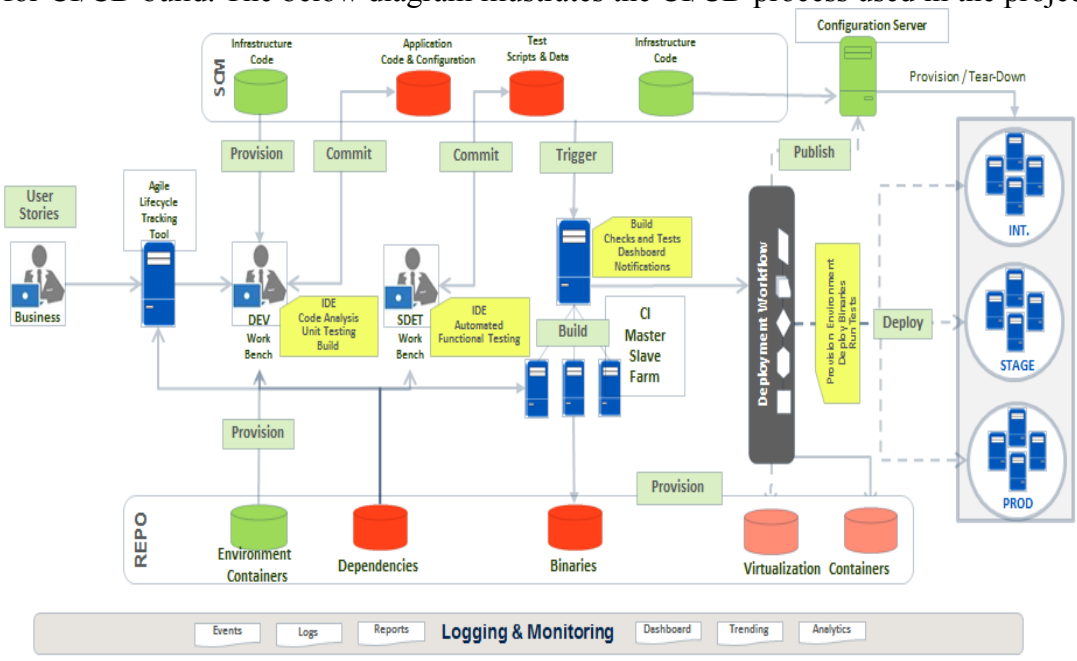
B. UAT Signoff by Product owners and business stake holders

C. Go-No-Go approval for final deployment

No of User Stories	System Testing Metrics			Regression Testing Metrics			User Acceptance Metrics		
	No of Test Cases executed	No of Test Failed	No of Tests Passed	No of Test Cases executed	No of Test Failed	No of Tests Passed	No of Test Cases executed	No of Test Failed	No of Tests Passed
6	72	23	72	49	11	49	21	3	21

3.5 Project CI/CD Pipeline

Leverages TFS for project tracking and GitHub for code maintenance and AWS code pipeline for CI/CD build. The below diagram illustrates the CI/CD process used in the project delivery



4 Tool Evaluation (All)

This section outlines the evaluation and analysis of tools used and studied.

4.1 Requirement Analysis Tool (Sarah)

I designed the mobile prototype using the JustinMind prototyping tool. It is a free tool to use, though it does require a download onto your computer. It's been used by companies like Adobe, Google, the IRS, Verizon, and Sony. It works kind of like Photoshop, only for prototyping. It is very user-friendly, with lots of videos and helpful tips associated with the different features. It has different templates available for use, pre-loaded UI kits, and templates that are customizable. It also offers testing for user experience. A feature of Justinmind that I didn't get to explore was the ability to publish prototypes and work on shared prototypes. The program seems like it would Segway nicely into a company that is already practicing an Agile environment, as the sharing capabilities means you could loop in clients sooner in the design process to receive feedback and make changes. The program has subscription options with additional features on two different levels. The company also offers perpetual licenses for companies with a cost of $455 per user on the first tier and $795 per user on the second tier. If there is a need for a collaboration server, the company offers that as well for companies that have sensitive product information and need the extra level of control. It is a tool I would like to spend more time exploring and a tool that I would recommend for anyone who needed to design a prototype quickly.

4.2 Project management Tool (Uma)

- Team Foundation server – This tool acts as Agile life tracking tool, CI/CD pipeline execution from dev to Prod, testing and defect tracking. Integration with Visual studio since our code base is in .NET
- GitHub – Hassle free yet cost effective Code management tool, built in code review tools and has seamless integration with AWS cloud component deployment model and SDK support
- AWS Code pipeline / Build – Leverages Amazon's managed service to run chef/Jenkins based CI/CD recipes for code deployment

4.3 Platform Selection (Uma)

The main goal of the platform selection is to "Use only What you need" and "Fit your payment model to your business model"

- Infrastructure Selection - Amazon Web Services offers cloud web hosting solutions that provide businesses, non-profits, and governmental organizations with low-cost ways to deliver their websites and web applications in a highly secured way by using HIPAA compliant services
- Database Tier – Leveraged AWS managed service database called Amazon RDS MySQL to store the data and report. No Licensing cost unlike the other commercial engine. Has in-built scaling option

- Application Tier – Leveraged AWS EC2 instances that has auto scaling and elastic load balancing to provide resiliency, reliability and fault tolerant virtual machines to deploy websites

4.4 Development Tools /IDE (Ahmed)

For this application, we are using the traditional approach of web development consists of writing HTML for the webpage with its own CSS and JavaScripts for the front-end part. And for the back-end and deployment process, we will be using Laravel framework, it simplifies the production process and takes much of the pain out of web app project. It will be used for tasks such as authentication, routing, containerization and queues. Laravel has its own migration system for manipulating database queries.

5 Lesson Learnt (All)

People	Process	Technology
Lesson / Risk • Domain knowledge is not well understood • Data Authorization not well explained • Limited Knowledge of Cloud platform • Internal SME/ development team availability for • No partner involvement	**Lesson / Risk** • Existing Patient Journey Vs New Patient journey for key business metrics • Timely availability of reporting needs • Lack of Data governance	**Lesson / Risk** • Communication between the team was challenge • Not much ramp up time given to the team to adapt cloud technologies • Code repository deploying cloud components was sustainable • Security review meetings esp data on cloud delayed development

Retrospective Measure
• Recommended Training
  • Health care Industry domain
  • Security
  • Cloud Platform
• Data security requirement well captured by working with Product owners / end users
• Functional knowledge walkthrough with dev team and Product team
• Consulting with partner network for cloud deployment to reduce time to market
	Retrospective Measure
	• RACI matrix Creation to resolve data governance to know who is accountable Vs Informational
	• Dev Team to work with Product team on the report mock ups and report validation
	• SLA should be mapped to each critical reports sent to Physician
		Retrospective Measure
		• Team started involving Cloud providers for detailed documentation on building their services
		• Detailed design review conducted with cloud providers before implementation
		• Start simple code pipeline and maintenance by adopting infrastructure as a code devops model
		• Security requirements should be done in preliminary or earlier phase rather tail of projects

6 Conclusion (All)

This section outlines the lesson learnt from Business as well as IT Perspective

Business Perspective (Sarah):

The market for Symp+rack is present. It is a product that will see use and can make a difference, especially with later iterations that link to fitness trackers to provide an overview of health. Pricing the application is where the real lessons were learned. In the middle of the project, the information about Facebook and data sharing broke and the general public began to be that much more aware of their data and privacy. Originally envisioned as a free to use application, a new freemium model had to be considered and certain aspects of the application needed to be redesigned, such as the limits to sharing care teams and how many symptom trackers a user could track for free.

IT Perspective (Uma):

There are numerous architectural and conceptual considerations when we were contemplating building Symp+rack web application to the AWS Cloud. The benefits of having a cost-effective, highly scalable, and fault-tolerant infrastructure that grows with your business far outstrips the efforts of migrating to the AWS Cloud.

Security Perspective (Ahmed):

Being an HIPAA compliant data, this product leverages data encryption service while the data is at-rest and in-transit through AWS default Key management service. Besides that, the application is implemented with Microsoft Active Directory solution on AWS for Identity Management service to authenticate the users and manage the audit-trial of the application.