Project Name: MEDIWORLD

Created By:

Sourabh Namilikonda

Hrishikesh Rendalkar

Pratik Dhumal

Avleen Singh Khanuja

UNIVERSITY: SAN JOSE STATE UNIVERSITY, San Jose, CA

COURSE: CMPE 281- Cloud Technologies

Guided By: Prof. Sanjay Garje

ISA- Divyankita Urs

LinkedIn:

Sourabh Namilikonda: https://www.linkedin.com/in/sourabh-namilikonda-255b226a/

Hrishikesh Rendalkar: https://www.linkedin.com/in/hrishikesh-rendalkar-5a6b17152/

Pratik Dhumal: https://www.linkedin.com/in/pratik-dhumal-2937aa72/

Avleen Singh Khanuja: https://www.linkedin.com/in/avleen-singh-khanuja-3ab637128/

Project problem statement

Large scale hospitals have the necessary IT infrastructure and resources to have a considerable online presence, but individual doctors and clinics don't have the expertise or investments for the same. The requirement is of a central portal where small scale doctors and patients can interact with equal facilities. Our application contains general as well as user specific customized information distribution in the Healthcare and Medical domain. We wish to build a repository of terms, procedures, organizations, names, definitions etc. to mine and provide relevant details to a user. Our aim is to simplify and provide accurate information which might be easier for a layman to understand.

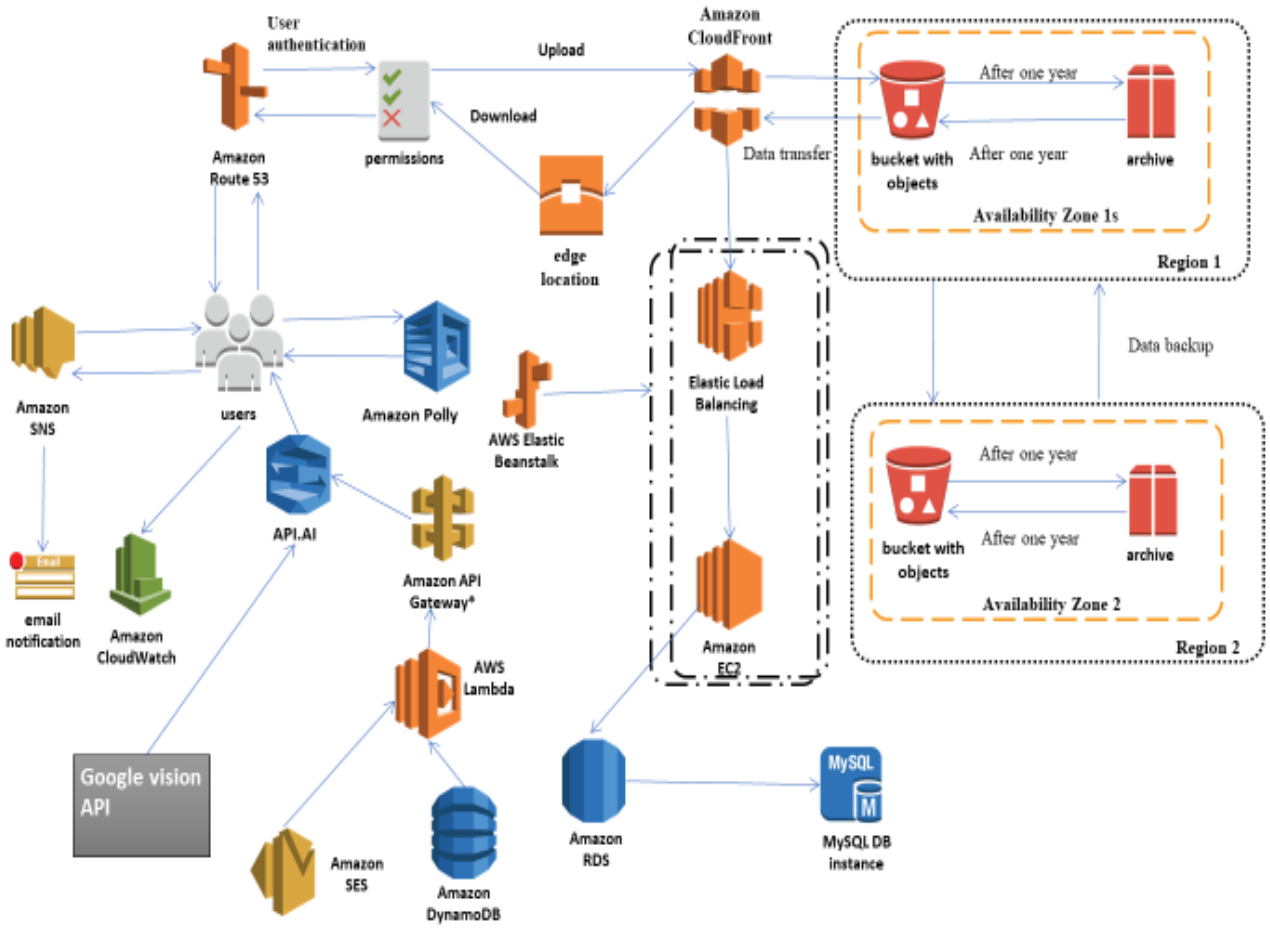
Project solution

Using various AWS services and Google cloud platform we have developed a 3-tier web application. We are using Chatbot like functionality where user will be able to resolve his queries in a fast and easy manner. Implementing image recognition will help user save their time by directly scanning an image and then getting the required text present in the image instead of typing long set of words. Patient will be able to upload their reports as and when they want. These reports will be available to them globally and at any given instance of time. Feature such as SMS is also implemented as a part of the project.

Features:

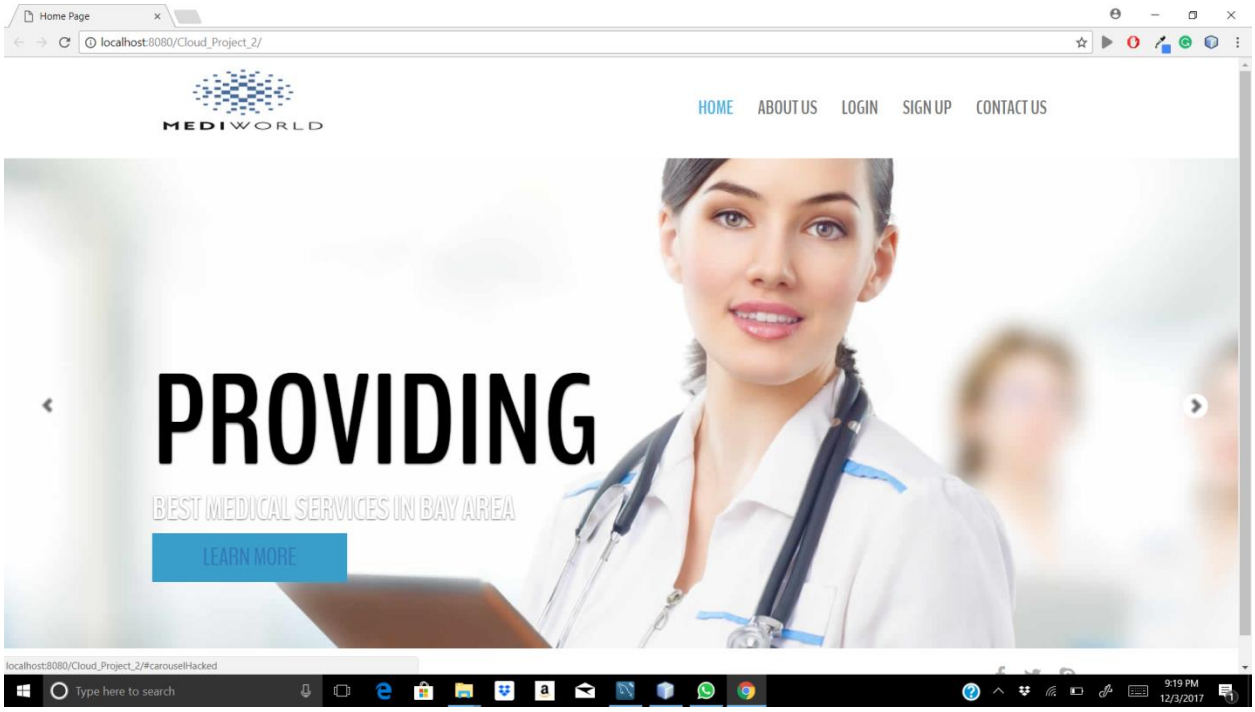
1. Register and login the user using registration form and login credentials. The user information will be stored in AWS RDS.
2. Upload, download and delete reports via portal. To perform these actions we are making use of AWS S3. We have configured life cycle policy, replication policy and transfer acceleration policy along with limited access.
3. We have implemented Chatbot using API.AI (dialog flow), AWS Lambda, API gateway, DynamoDB and SES
4. We have implemented AWS Polly for notifications of different actions performed by the user such as after login a welcome message will be audible. We have also implemented this functionality when the user uploads, downloads and deletes the report.
5. User can view scheduled appointment from the portal. To achieve this we have used AWS DynamoDB.
6. Image Recognition is implemented using Google vision API. Here text written in the image will be retrieved and search functionality will be executed against the retrieved text.
7. Content Search is implemented using Wikipedia and YouTube API to get all information from different location at one place.
8. Set reminders functionality is implemented using AWS cloudWatch and SNS. User can set reminders at specific time interval.

Architecture Diagram:

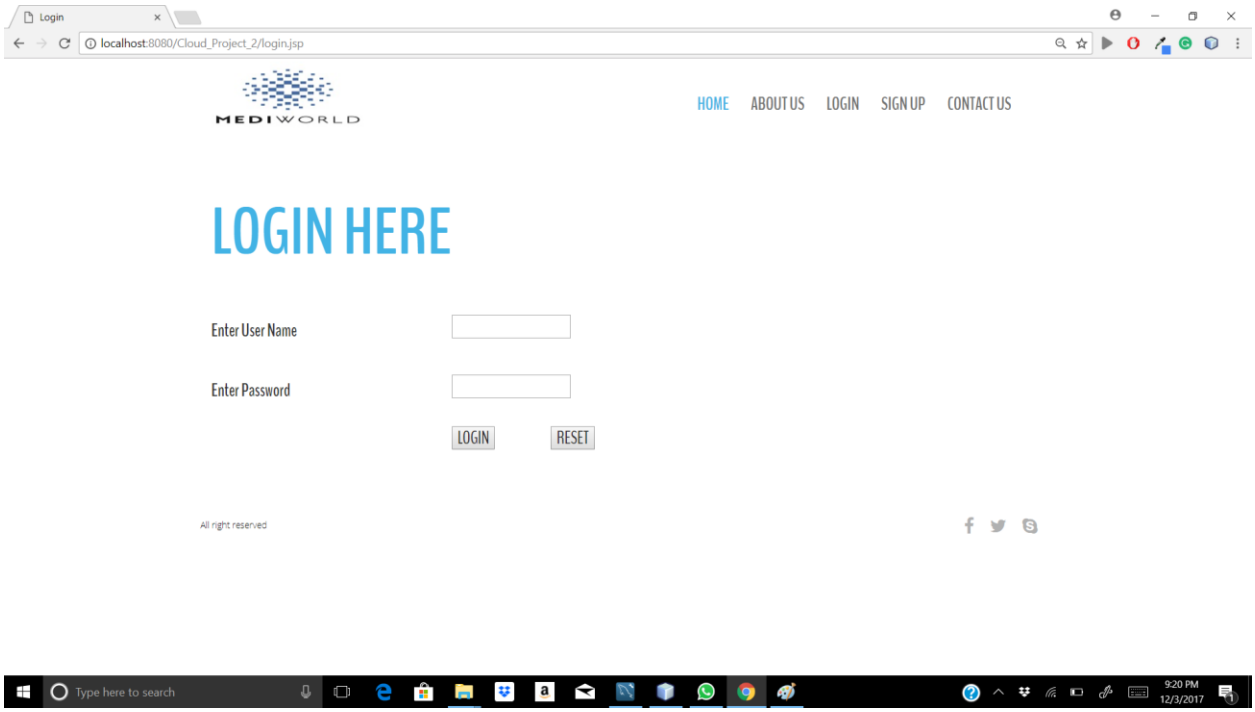


Demo Screenshots:

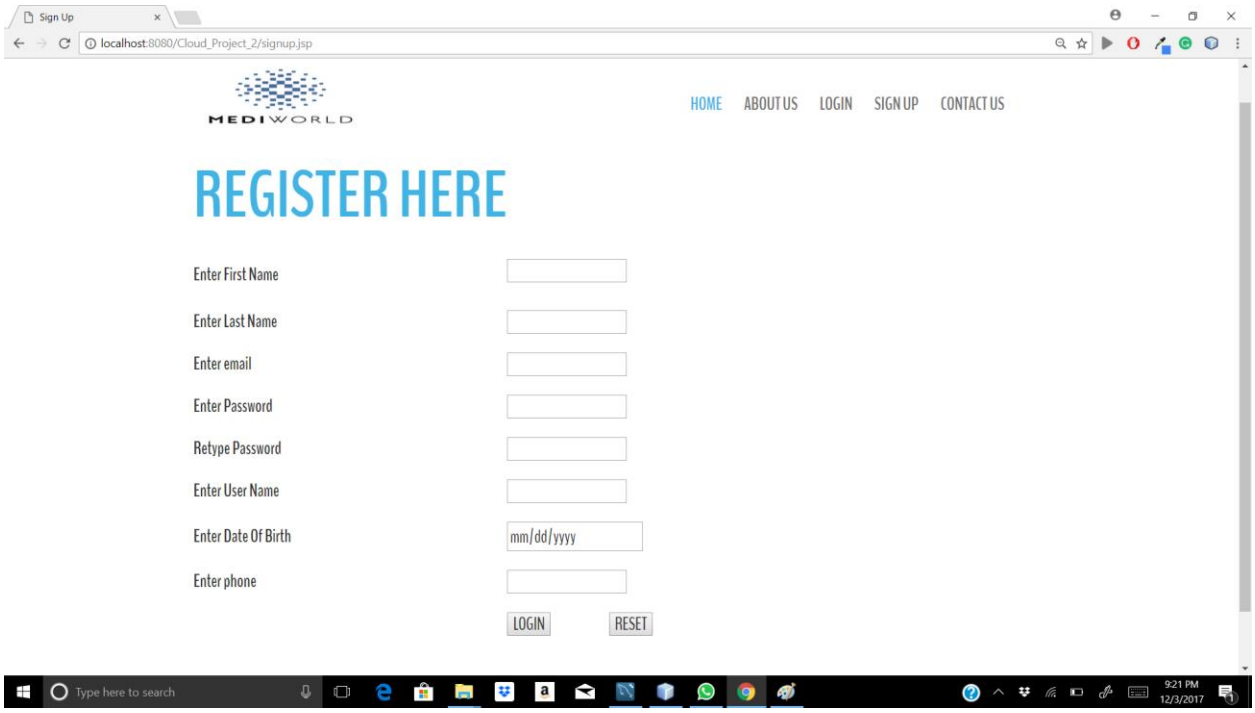
- Home Page



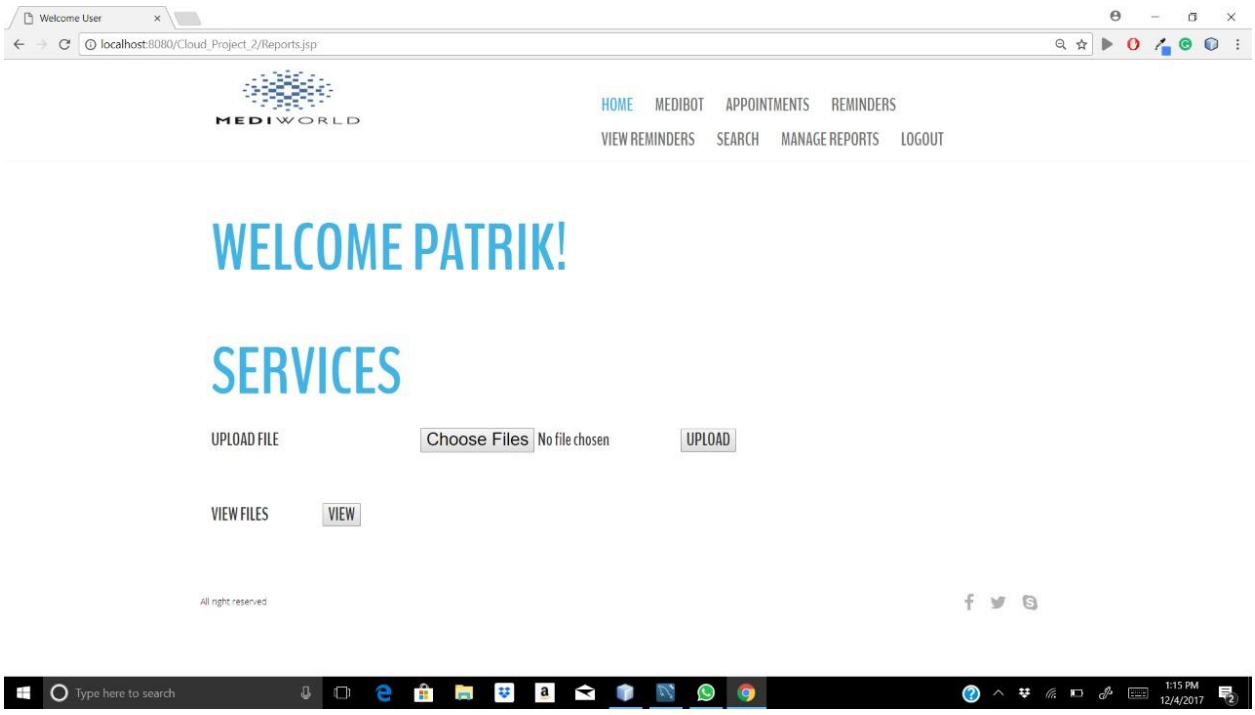
- Login



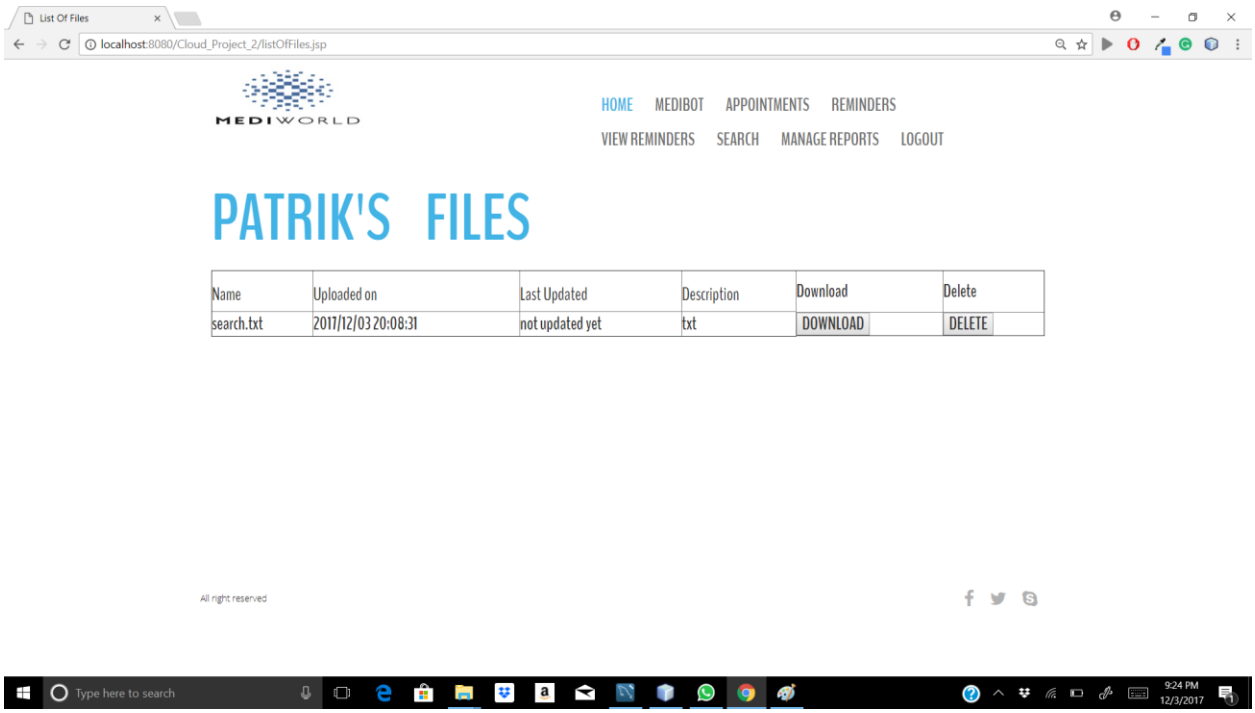
- Signup



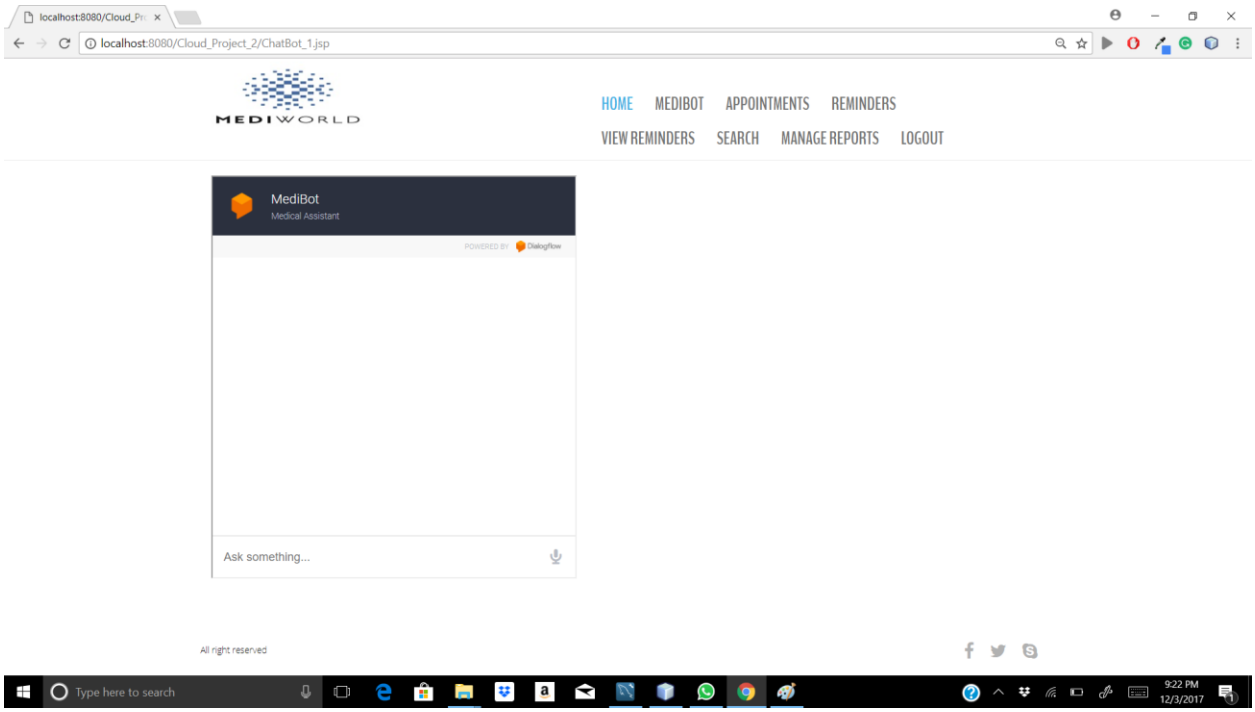
- Upload Reports



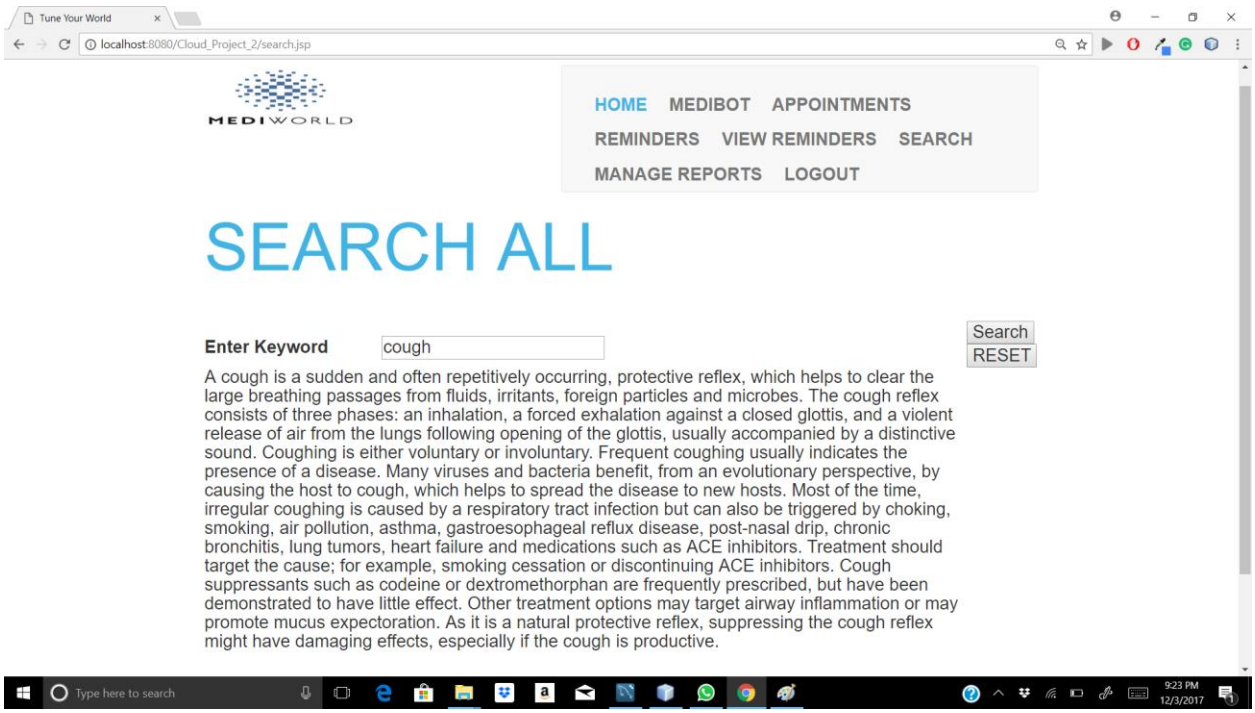
- List Reports



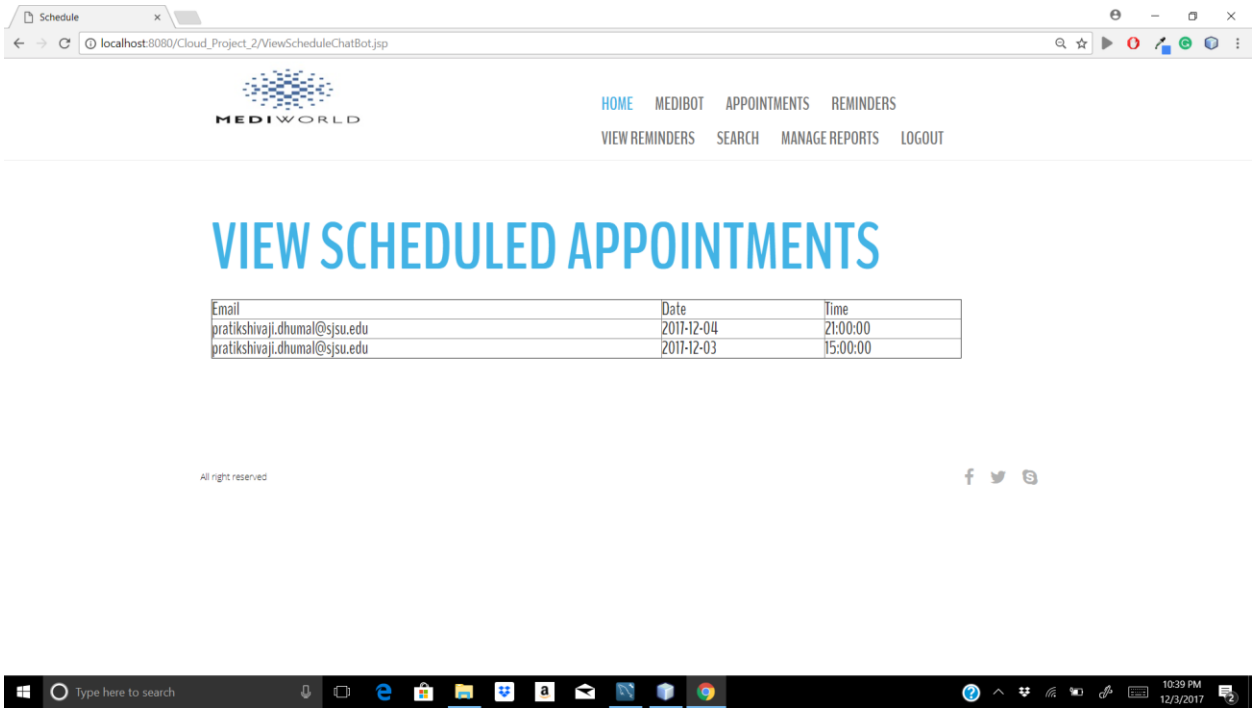
- MediBot



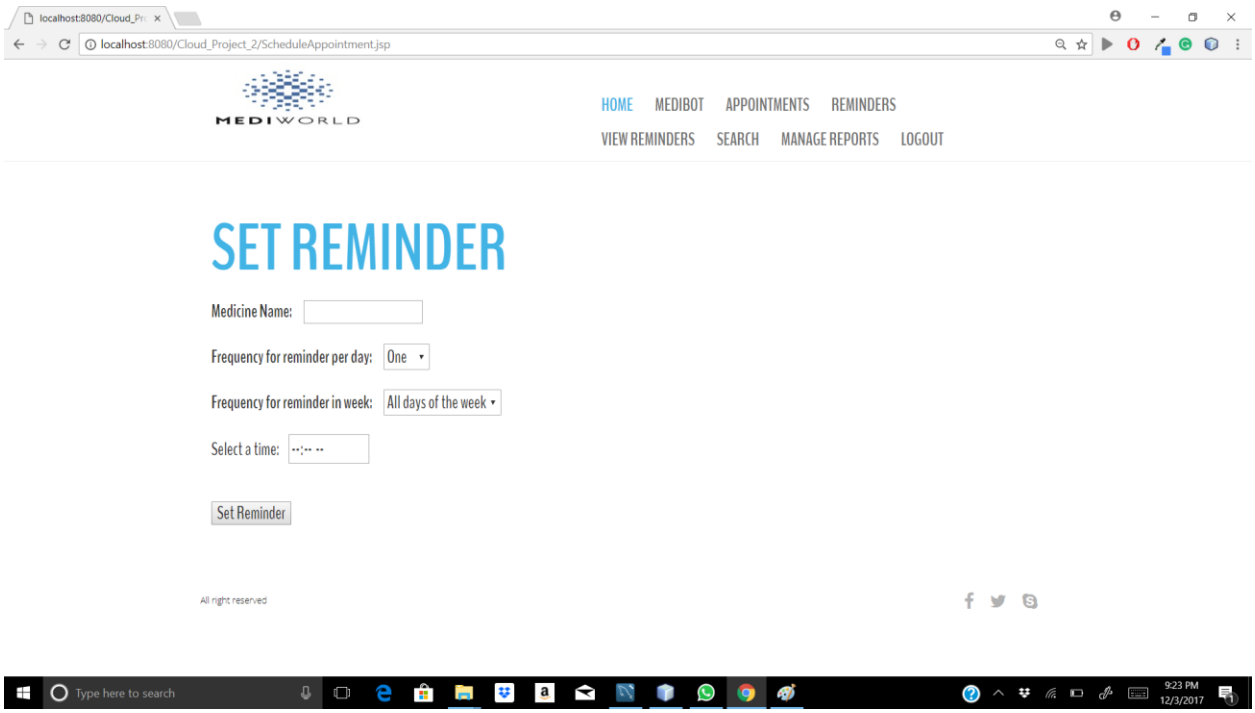
- Search Content



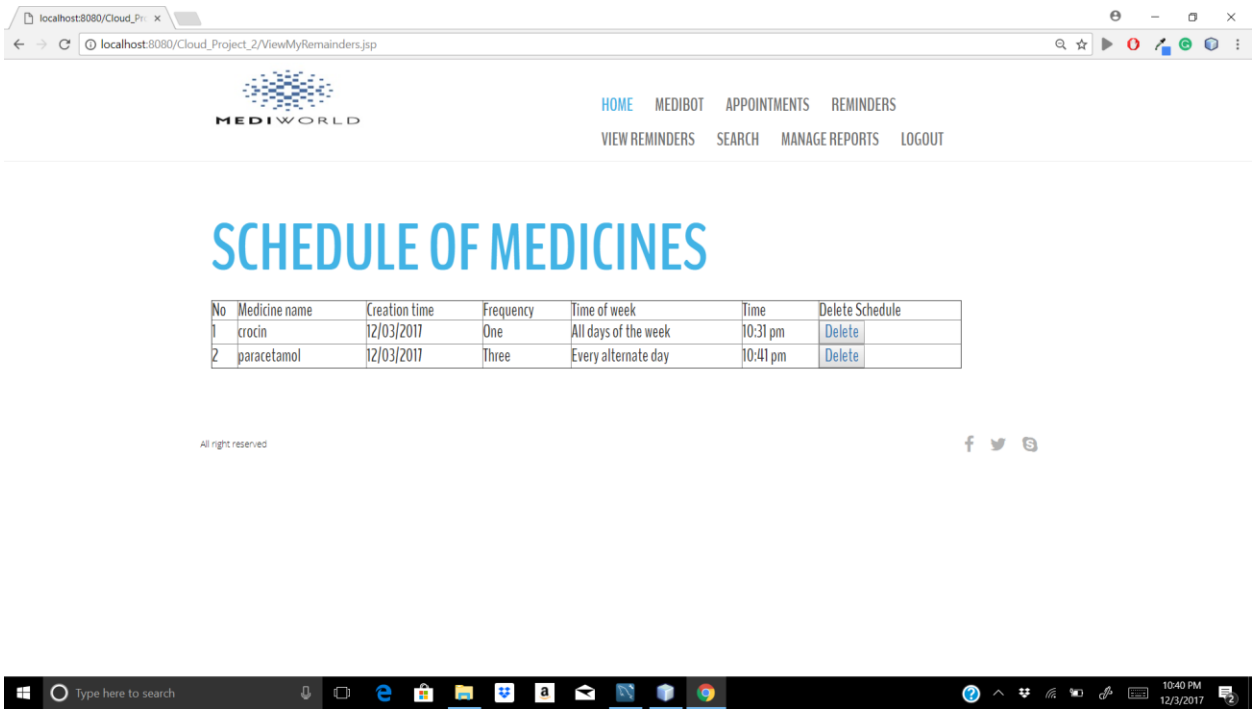
- Scheduled Appointments



- Set Reminders



- View Reminders



Pre Requisite Configurations:

Configure and setup the following Services of Amazon AWS by creating an account at https://aws.amazon.com/ and then setting up:

1. Amazon AWS Elastic Beanstalk
2. Amazon AWS S3
3. Amazon AWS S3 Transfer Acceleration
4. Amazon AWS CloudFront
5. Amazon AWS IAM
6. Amazon AWS EC2
7. Amazon AWS Route 53
8. Amazon AWS RDS
9. Amazon AWS ELB
10. Amazon AWS Auto Scaling Groups
11. Amazon AWS SNS
12. Amazon AWS Lambda
13. Amazon AWS CloudWatch
14. Amazon AWS SES
15. Google API.AI
16. Amazon AWS DynamoDB
17. Amazon AWS Polly
18. Google Vision API

List of Software Required to run the Application locally:

1. Java 1.7 (or higher) [JDK & JRE]

2. Netbeans IDE 8.2

3. MySQL Server 5.6

4. MySQL Workbench 6.2

5. Apache Maven 3.5.0

6. Any Browser(Google Chrome or Mozilla Firefox Preferred)

7. AWS SDK for Java version 1.11.215

Quick Steps:

- Install JAVA and set Environment Variables required and also install the JAR file for JDBC Connector.
- Install MySQL Server and then install MySQL Workbench.
- Install Apache Maven and set it's PATH.
- Install NetBeans IDE and create a new Maven Web Project thereby, implementing the JSP Pages, JAVA Classes and Servlets in the same and testing the code to resolve bugs.
- Create connection with JDBC Driver by configuring Access Credentials for Admin Properties which you'll be entering while installing MySQL Workbench.
- In MySql Workbench, create a new connection at localhost and port 3306 and create the required database and tables by running the SQL queries.
- This creates a JDBC and SQL connection with localhost.
- Install the Required Maven dependencies and download them.
- Download, clean and build the code in Netbeans IDE and run it.
- In case you want to deploy the code on the any other Web Server, you will need a WAR file to do so for which in the Windows Command Line, go to the folder where the project resides and run the following two commands one after the other:
  1) mvn compile
  2) mvn package

- This creates the WAR file which can be deployed on any Web Server.