Prabhav Jani

94014.prabhav@gmail.com **GitHub:** https://github.com/progressivePRV | **LinkedIn:** https://www.linkedin.com/in/prabhav-jani/

Education:

- University of North Carolina at Charlotte, North Carolina, USA
  Pursuing Master of Science in Computer Science
  Expected graduation Dec. 2021

- Gujarat Technological University, Ahmedabad, India
  Bachelor of Engineering in Computer Engineering
  Graduated in June 2019

Technical Skills

- Java
- SQL
- Express.js
- Html
- Firebase
- JavaScript
- Python
- AWS
- Android App Development

Achievements and Certificates:

- Red Hat Certified Engineer (RHCE) in 2019
- Awarded Certificate for "The Joy of Computing Using Python" from NPTEL Online Certification in 2018 (was among top 5%, all over India)
- Red Hat Certified System Administrator (RHCSA) in 2018

Experience:

- Graduate Teaching Assistant
  University of North Carolina at Charlotte
  Started: 01/2021 *(Current)*
  - Solve student's doubts regarding assignments and lecturers and grade their submissions. [course: Mobile Application Development]

Academic Projects:

- **Last Year Bachelor's Project** (Spring 2019)

- *Web Application Cluster without Shared Storage* (Pacemaker)(MySQL Galera)
  - Utilized Pacemaker with "Round-Robin" strategy for two web host servers.
  - Utilized MySQL Galera for maintaining redundancy in three database server.
  - In case of any server node shutting down unexpectedly, service will shift to redundant server and will not interrupt any running processes.

- **Computer Communication Projects** (Spring 2020)

- *HTTP server for Get and Put request* (JAVA)
  - Developed a Java Code which acts as server, which can server webpages/files using HTTP Get protocol
  - Developed a Java Server Code to handle PUT request.

- *Distance Vector Routing Protocol* (Python)
  - Developed python scripts which act as a router and uses Distant Vector Routing protocol to find shortest path to other routers in network.

- **Database System** (Spring 2020)

- Parking Management System (MySQL)
    - Designed database for the project, which includes table's structure, relationships, Indexes, and Normalization.
    - Implemented
        - View:
            - *todays_parking_tickets*: Gives information about all today's parking tickets
            - *vacant_spots*: Gives out all the vacant parking spots.
            - *monthly_revenue*: Gives output of revenue generated in past months
        - Stored Functions:
            - *isVaccant*: Checks whether a particular spot is empty or not.
        - Stored Procedure:
            - *Spot_Information*: It will provide details like user name, license plate number of the vehicle which is parked at a particular spot.
            - *Vacancy_in_Building*: It will provide a list of vacant spots in the building. When a user arrives at a building, it will be helpful in finding a spot.
        - Triggers:
            - <Before Insert> on <parking_ticket > **changeVaccancy**

              Whenever a new entry is provided in a parking ticket, it will check for the spot id vacancy. If it is vacant, it will enter the data into the parking ticket else error will be displayed stating the parking spot is not vacant.

            - <Recurring Event> **vaccancy_change**

              To change vacancy after check out time passes, this event would automatically change is_vaccancy to whichever spot has checked out

- **Mobile Application Development** (Spring 2020)

- News App (Java)
    - Utilized Async task to Consume a NEWS API for getting current new updates.
    - Read JSON response from the API and showed it using Recycler View.
    - Designed UI to show category of news article, related picture, heading and small description of news and "next" and "previous" symbol buttons for navigation.

- To-do App (Java)
    - Developed a code for sorting of Task on basis of Priority and Date.
    - Developed a code to take input from user to create Task with description, date and priority

- Trivia Quiz App (Java)
    - Developed a code to consume questions API
    - Designed UI to show question, related picture, options, "quit" button and "next" symbol button.
    - Implemented code to save answers of the user and show percentage correct (result) at the end of quiz.
    - Implemented timer to automatically submit the quiz and show the result.

- Mail App (Java)
    - Implemented code to consume REST APIs for sending and receiving mail
    - Developed an App to show received mail, compose mail and delete mail.
    - Implemented code to get information from user and register user using it.

- Weather App (Java)
    - Developed an App to show weather of user selected cities.
    - Consumed "Accu weather" APIs for getting today's weather information of selected city, getting weather updates of next 5 days, and getting options for selection of city with same name and country. (example Charlotte, NC, US and Charlotte, MI, US)
    - Implemented functionality to save a city as a Favourite city for easy access of weather information about that city.

- Contact App (Java) (Firebase)
    - Developed a code to utilized mobile camera to take a picture and save it for a contact.
    - Created an App to store Contact details like First Name, Last Name, contact number, email and profile photo for specific logged in user using Firebase.

- Expense App (Java) (Firebase)
    - Created an App to store Expense details like Expense Name, Expense Category, Amount and Date in Firebase.
    - Implemented code to edit expense in the Firebase.
    - Implemented code for deleting expense from Firebase.

- Trip planning App (Java) (Firebase) (Google API's)
    - Used nested Recycler View to show trips and include place to visit during the trip.
    - Utilized Google's "Search For city" API for suggestion of cities while user write a city name.
    - Utilized Google's "Get Places close to location" API to suggest places to visit in the city.

- **Advance Mobile Application Development** (Fall 2020)

- Chatroom (Java) (Firebase)
    - Implemented User Database in Firebase
    - Implemented Navigation Drawer menu with fragment navigation.
    - Developed code for updating user profile

- Share-a-ride APP (Java)
    - Programmed App for listening live driver response for requested ride.
    - Implemented code for listening to live driver's location.
    - Engineered a solution code for saving ride information into the database with automatic detection of proximity between driver and the location.

- Uno Game APP (Firebase)
    - Developed Game UI, created game card backgrounds and deck.
    - Made cards responsive, adjusted different characters into the card. (i.e. skip, +4 and 0-9)
    - Used Recycler view to show in hand (user's) cards.

- Shopping App (Java) (Braintree SDK) (Express.js)
    - Integrated Braintree SDK in App, for Payment using credit card or debit card.
    - Implemented code for saving cart information and showing shopping history using nested Recycler View.
    - Implemented API calls for storing and getting shopping history.

- Sales Record App (Java)(Room)(Express.js)
    - Implemented Tab Layout for favourite record and All sales record.
    - Implemented Room on device database to store favourite sales.
    - Designed on device database to store user specific favourites.
    - Developed code for getting live updates of favourites.

- Auction App (Java)(Firebase)
    - Organized parts of App UI referring Material Design.
    - Implemented Bottom Tab Layout and fragment Transaction Manager
    - Developed code to handle Foreground Notification.
    - Done App testing

- Examiner App (Java)(NodeJS)
    - Programmed mobile Camera to detect QR-code using CameraX (Jetpack library) and Barcode Scanner (ML kit).
    - Utilized OkHttp to call the questions API.
    - Created animation for flow to Next and Previous Questions.

- LiveLIVE Security Streaming Service (LiveSwitch SDK)(Express.js)
    - Implemented LiveSwitch SDK.
    - Utilized LiveSwitch SDK to capture the camera and microphone feed.
    - Utilized LiveSwitch SDK to join and leave broadcast channels.
    - Utilized LiveSwitch SDK for creating and destroying upstream and downstream connections.
    - Developed a helper class to integrate LiveSwitch SDK's functionality into both the Apps.

- **Algorithm and Data Structure** (Fall 2020)

- Comparison-based Sorting Algorithms (Python)
    - Implemented Insertion Sort
    - Implemented Merge Sort
    - Implemented Heap Sort
    - Implemented Quick Sort
    - Implemented Modified Quick Sort
    - Developed a code to randomly generate 5 set of integer arrays for each size in set => [1000,2000,3000,4000,5000,10000,20000,30000,40000,50000]. 5 set to get an average of time taken.
    - Calculate the time taken by each algorithm to sort arrays of different size and at the end show graph of time taken by each algorithm using Matplotlib.

- Dijkstra's Algorithm [Single Source Shortest Path] (python)
    - Implemented Dijkstra's Algorithm to find shortest path from one source node to other nodes.
    - Implemented Adjacency List for efficiency (reducing time complexity) of the algorithm.
    - Used array-based Heap as priority queue.
- Prim's Algorithm [Minimum spanning Tree] (python)
    - Implemented to find the minimum spanning tree from the given graph.
    - Implemented Adjacency List for efficiency (reducing time complexity) of the algorithm.
    - Used array-based Heap as priority queue.