

A
Project Report
On
CHILD VACCINATION TRACKER
Submitted by
Akhilesh Mangesh Sathe
Roll No.: 22356
MCA–I
SEM–I
Under the guidance of
Prof. Meenakshi Jadhav
For the Academic Year 2022-23



Sinhgad Technical Education Society's
Sinhgad Institute of Management

Vadgaon Bk Pune 411041

(Affiliated to SPPU Pune & Approved by AICTE New Delhi)

Prof. M. N. Navale
M.E. (ELECT.), MIE, MBA
FOUNDER PRESIDENT

Dr. (Mrs.) Sunanda M. Navale
B.A., MPM, Ph.D
FOUNDER SECRETARY

Dr. Chandrani Singh
MCA, ME, (Com. Sci.), Ph.D
DIRECTOR - MCA

Date:

CERTIFICATE

This is to certify that Mr Akhilesh Mangesh Sathe has successfully completed his project work entitled “**Child Vaccination Tracker**” in partial fulfillment of MCA – I SEM –I Mini Project for the year 2022-2023. He has worked under our guidance and direction.

Prof. Meenakshi Jadhav
Project Guide

Dr. Chandrani Singh
Director, SIOM-MCA

Examiner 1

Examiner 2

Date:

Place: Pune

Celebrating 25 Years
OF ACADEMIC EXCELLENCE

DECLARATION

I certify that the work contained in this report is original and has been done by me under the guidance of my supervisor(s).

- The work has not been submitted to any other Institute for any degree or diploma.
- I have followed the guidelines provided by the Institute in preparing the report.
- I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.
- Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the report and giving their details in the references.

Name and Signature of Project Team Members:

Sr. No.	Seat No.	Name of students	Signature of students
1		Akhilesh Mangesh Sathe	

ACKNOWLEDGEMENT

It is very difficult task to acknowledge all those who have been of tremendous help in this project. I would like to thank my respected guide **Prof. Meenakshi Jadhav** for providing me necessary facilities to complete my project and also for their guidance and encouragement in completing my project successfully without which it wouldn't be possible. I wish to convey my special thanks and immeasurable feelings of gratitude towards **Dr. Chandrani Singh, Director SIOM-MCA**. I wish to convey my special thanks to all teaching and non-teaching staff members of **Sinhgad Institute of Management, Pune** for their support.

Thank You

Yours Sincerely,

Akhilesh Mangesh Sathe

INDEX

Sr. No.	Chapter	Page No.
1	CHAPTER 1: INTRODUCTION	
1.1	Abstract	
1.2	Existing System and Need for System	
1.3	Scope of System	
1.4	Operating Environment Hardware and Software	
1.5	Brief Description of Technology used	
2	CHAPTER 2: PROPOSED SYSTEM	
2.1	Feasibility Study	
2.2	Objectives of the proposed system	
2.3	Users of the system	
3	CHAPTER 3: ANALYSIS AND DESIGN	
3.1	Entity Relationship Diagram (ERD)	
3.2	Class Diagram	
3.3	Use Case Diagrams	
3.4	Activity Diagram	
3.5	Sequence Diagram	
3.6	Component Diagram	
3.7	Module and Hierarchy Diagram	
3.8	Table Design	
3.9	Data Dictionary	
3.10	Sample Input and Output Screens (Screens must have valid data. All reports must have at-least 5 valid records.)	
4	CHAPTER 4: CODING Sample code	
5	CHAPTER 5: LIMITATIONS OF SYSTEM	
6	CHAPTER 6: PROPOSED ENHANCEMENTS	
7	CHAPTER 7: CONCLUSION	
8	CHAPTER 8: BIBLIOGRAPHY	

CHAPTER 1: INTRODUCTION

2.1 Abstract

Vaccines play a crucial role in preventing life-threatening diseases, especially in children as their immune system isn't fully developed. Timely immunization is essential for children to protect them from life-threatening diseases that vaccines can easily prevent.

Child Vaccination Tracker is a web application with the goal to help parents to keep track of the vaccinations of their child. It gives the parents timely notifications reminding them of the upcoming vaccination of their child, this ensures that child receives vaccines as per the schedule. It serves the parents with essential information regarding the vaccinations such as the disease prevented, advantages, possible side effects of the vaccine.

It lets the parents book an appointment with vaccination center that are registered with the application, also it lets the parents to locate the vaccination centers with ease. Vaccination centers can register with the app and create their profile to display their timings, location and other details.

2.2 Existing System and Need for System

There exist many applications which allow parents to track the growth of their child along with the vaccinations tracking but they focus mostly on the growth tracking aspect rather than vaccination.

Child Vaccination Tracker focuses on the vaccinations of the child, providing the parents with necessary information, references and statistics regarding the same. Parents can be free of worries about their child's vaccinations as the app reminds them about the upcoming vaccinations. The application also becomes the platform for parents and registered vaccination centers to meet.

2.3 Scope of System

Children need to be immunized against various diseases, illnesses caused by viruses and bacteria which on a broader scale prevents epidemics.

This web application is designed to help parents to get their children vaccinated as per the immunization schedule.

Scope of this web application is as follows:

- This web application will let parents to track the vaccination of their child.
- This web application will maintain user data.
- It will provide the parents with timely reminders about vaccinations.
- It will let provide parents with vaccination information for each vaccine.
- Vaccination centers will be able to register and display their information and timings.
- Parents will be able to book appointments with the registered vaccination centers.
- Statistics will be generated for users and vaccines.
- Parents will be able to locate vaccination centers near them.

2.4 Operating Environment Hardware and Software

Hardware Requirements

- Processor: Pentium IV or above
- RAM: 1 GB or above
- HDD:5 GB or above

Software Requirement

- Operating System: Windows, Linux or MacOS
- A browser which supports JavaScript

2.5 Brief Description of Technology used

- Front End: HTML, CSS, JavaScript, React.JS
- Back End: Express.JS, Node.JS
- Database: MongoDB

CHAPTER 2: PROPOSED SYSTEM

2.1 Feasibility Study

A feasibility study for a vaccination system would assess the potential success and risks of developing and implementing a vaccination tracking and reminder system. The study would typically involve the following components:

1. **Market feasibility:** Assessing the need and demand for a vaccination tracking and reminder system among parents and healthcare providers. This would involve analyzing factors such as the number of children and families in the target market, the current vaccination rates, and the availability of competing vaccination tracking systems.
2. **Technical feasibility:** Evaluating the technical requirements and constraints of the system. This would include assessing the necessary hardware and software requirements, such as server space, database management systems, and application development tools.
3. **Financial feasibility:** Evaluating the financial viability of the project, including the initial investment required, ongoing expenses, revenue projections, and profitability.
4. **Legal and regulatory feasibility:** Assessing the legal and regulatory framework for the system, including compliance requirements for data privacy and security.
5. **Operational feasibility:** Evaluating the operational requirements and constraints of the system, including the availability of resources, workforce, and infrastructure needed to support the system.

Based on the results of the feasibility study, the developers can determine whether to proceed with the project, modify it to address potential risks and challenges, or abandon it if it is deemed unfeasible.

2.2 Objectives of the proposed system

- The objective of this project is to create a web application that will allow parents to keep track of the vaccinations of their child,
- Receive reminders for their child's next vaccination
- Book an appointment with vaccination centers.
- The web application will have the following key features
- Parents can access information regarding each of the listed vaccinations.
- They will have the ability to locate vaccination centers that is at the closest proximity.
- It will follow the National Immunization Schedule for infants and children.

2.3 Users of the system

1. Parent:

The parent is the user of the system who uses the system to get their children vaccinated. The parent can add more than one child and get appointments for nearest vaccination center.

2. Vaccination Center:

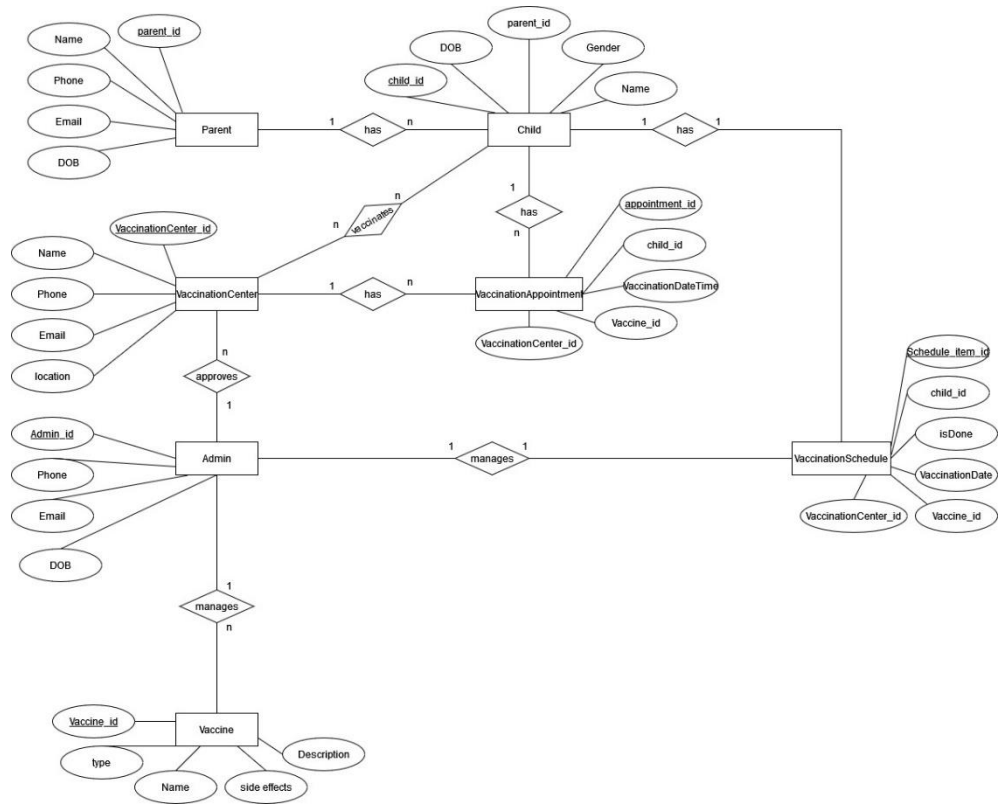
The vaccination center registers and lists available vaccines. Also Vaccination Center gives operational hours.

3. Admin:

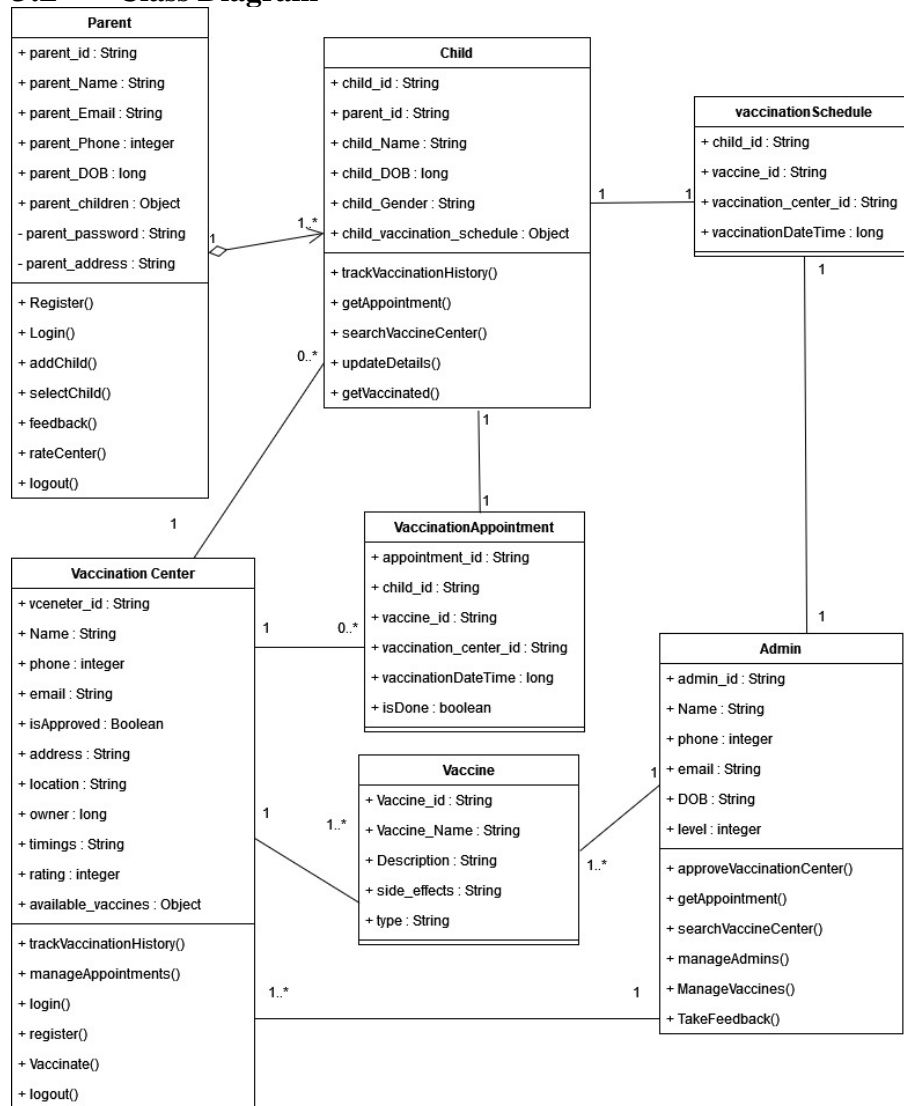
The admin is responsible for managing vaccines and vaccination centers.

CHAPTER 3: ANALYSIS AND DESIGN

3.1 Entity Relationship Diagram (ERD)

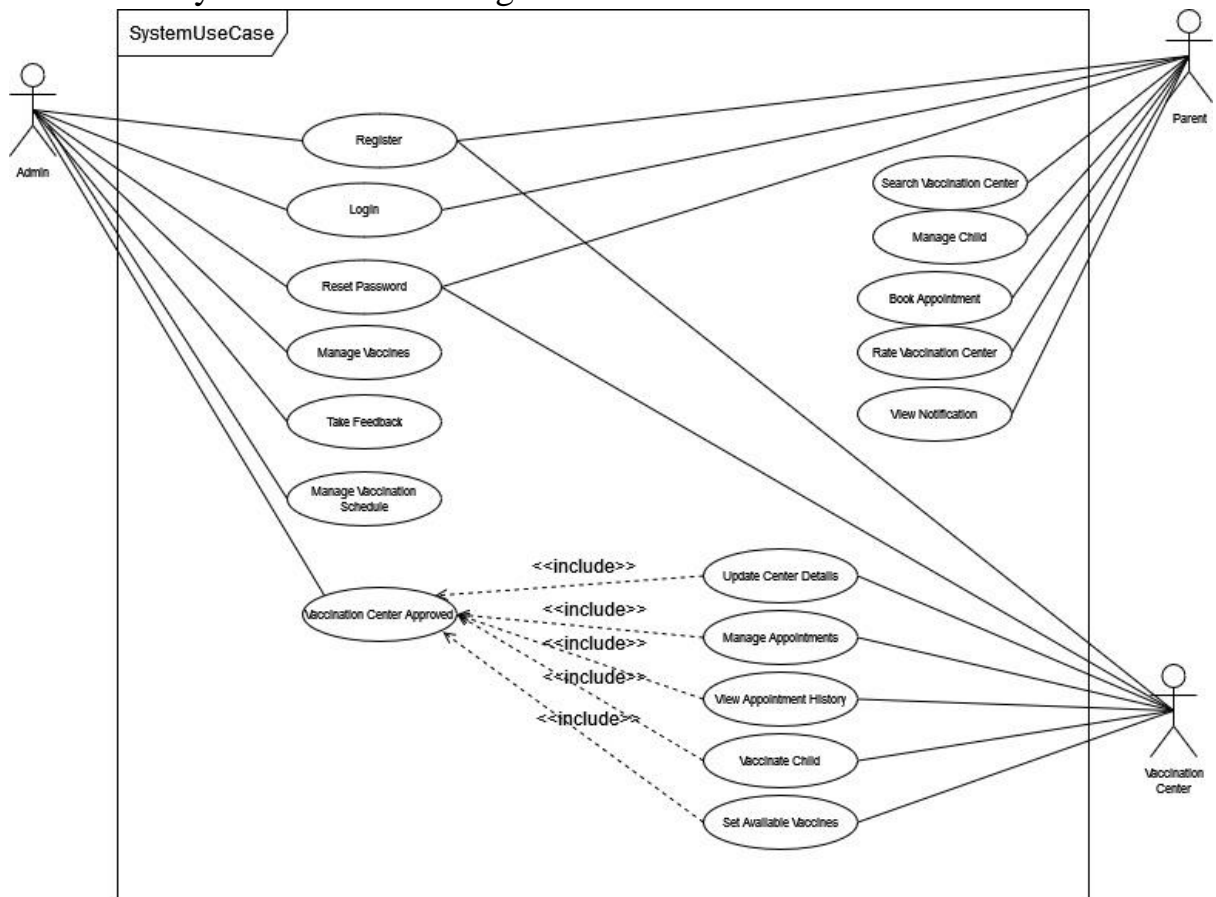


3.2 Class Diagram

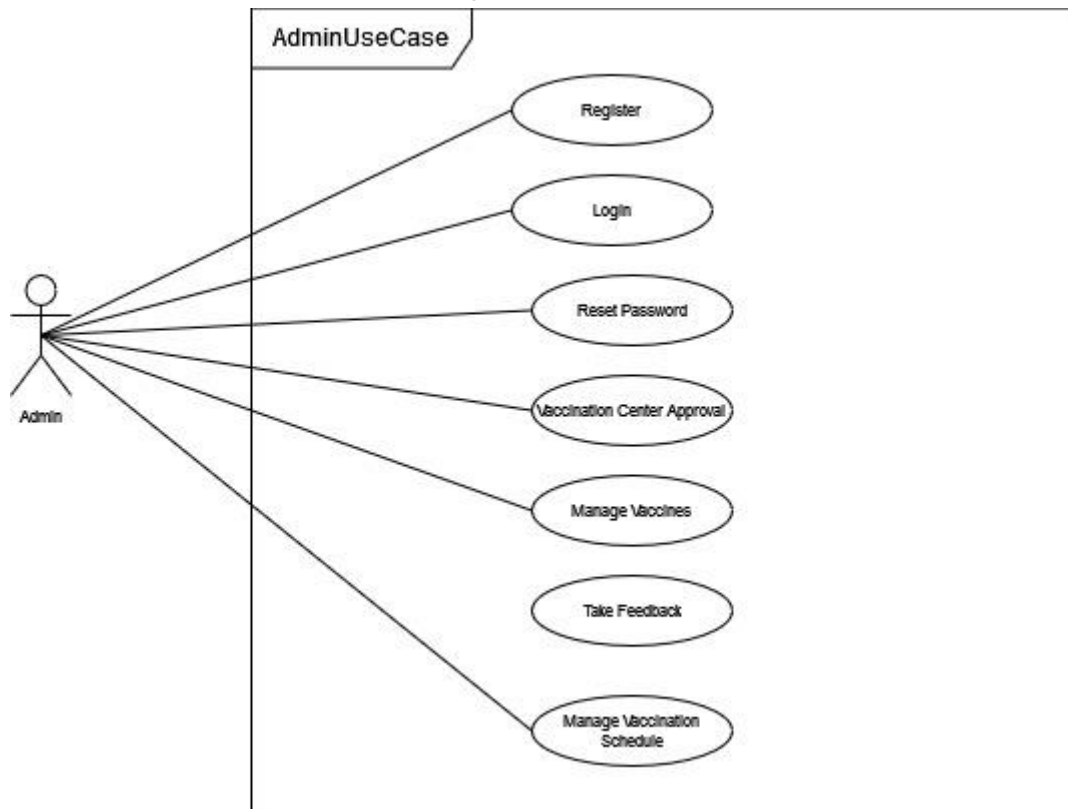


3.3 Use Case Diagrams

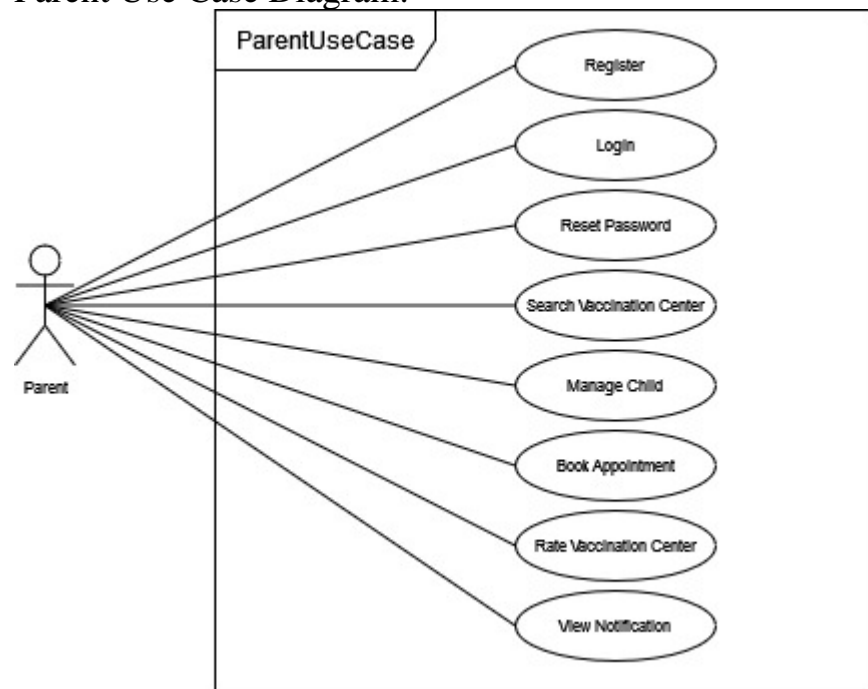
System Use Case Diagram:



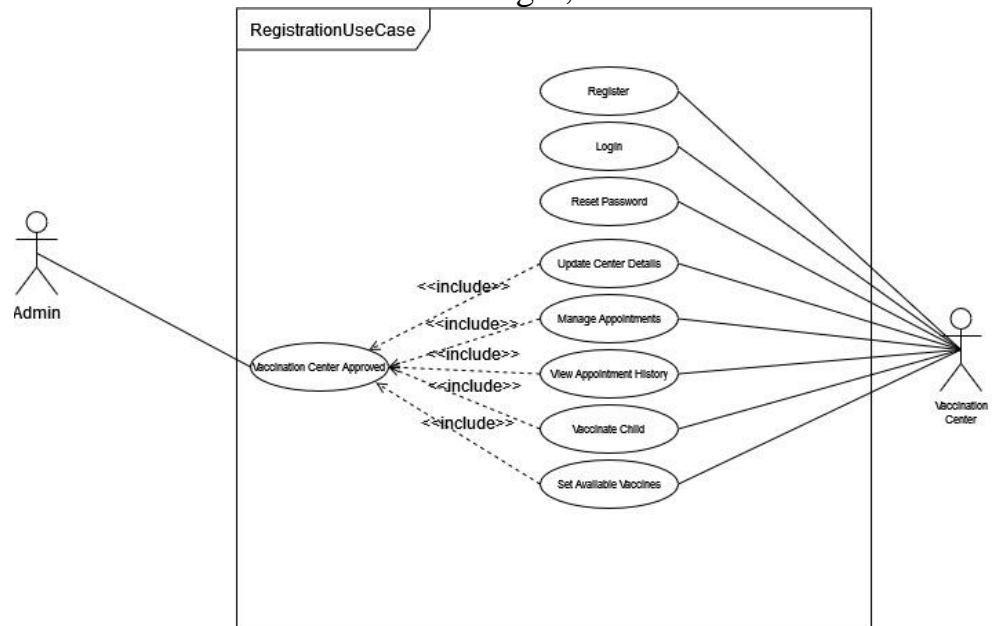
Admin Use Case Diagram:



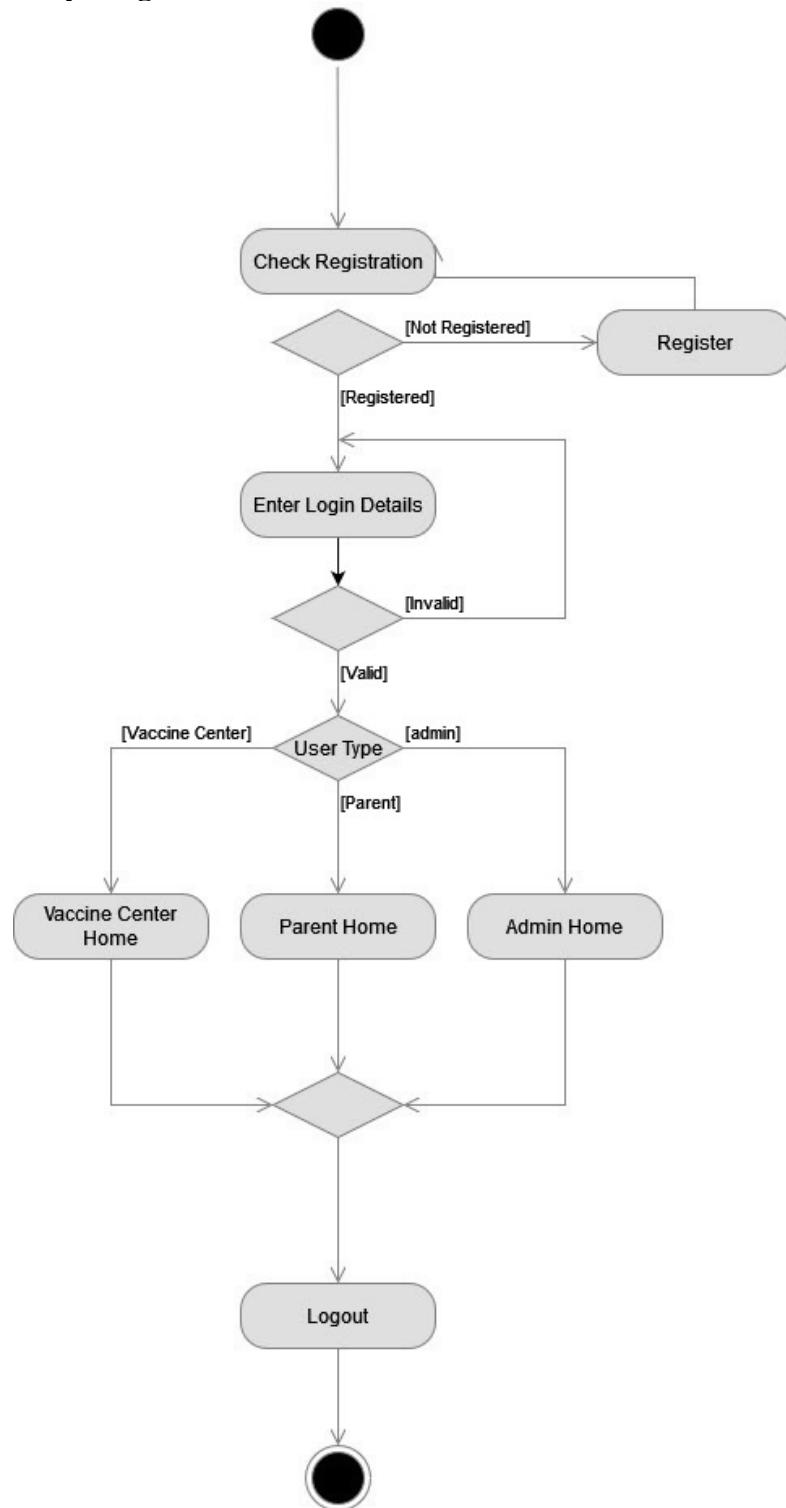
Parent Use Case Diagram:



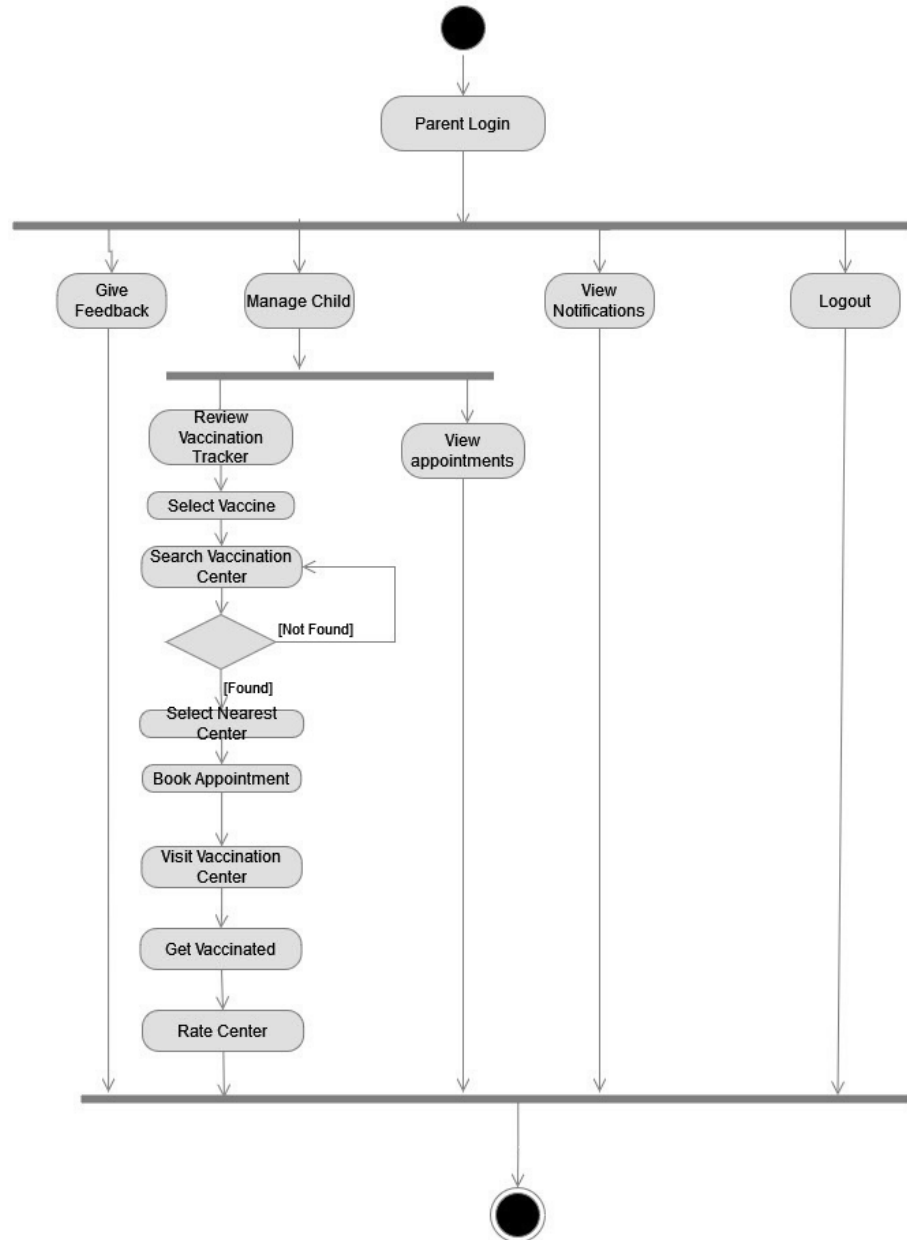
Vaccination Center Use Case Diagram,



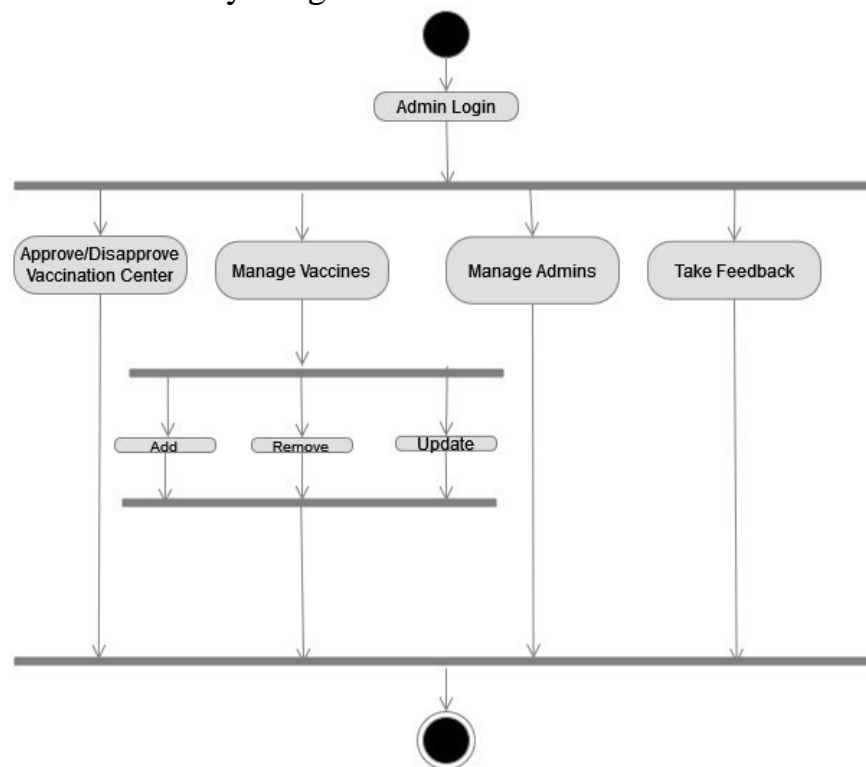
3.4 Activity Diagram



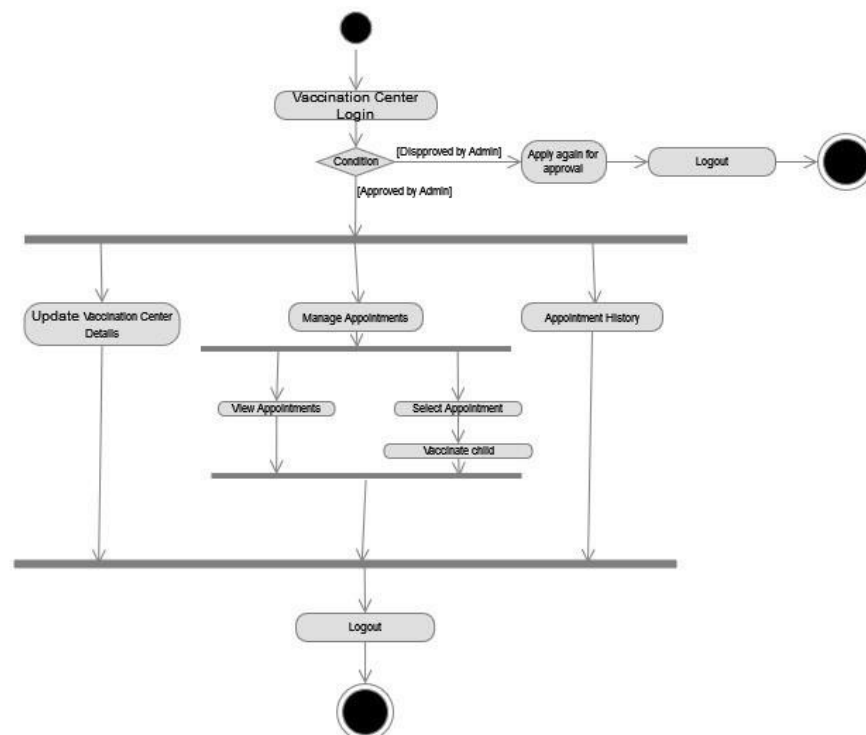
Parent Activity Diagram:



Admin Activity Diagram:

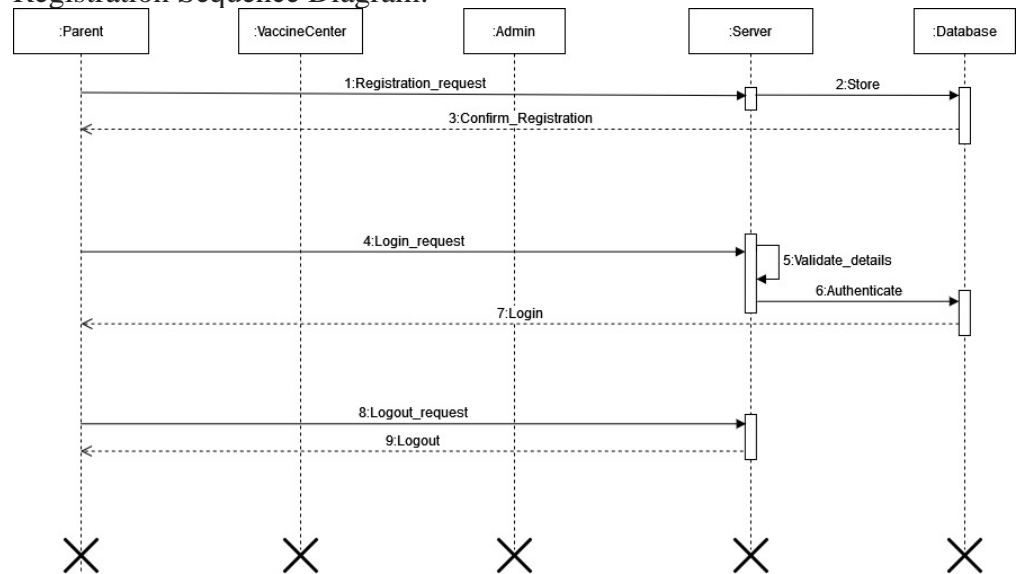


Vaccination Center Activity Diagram:

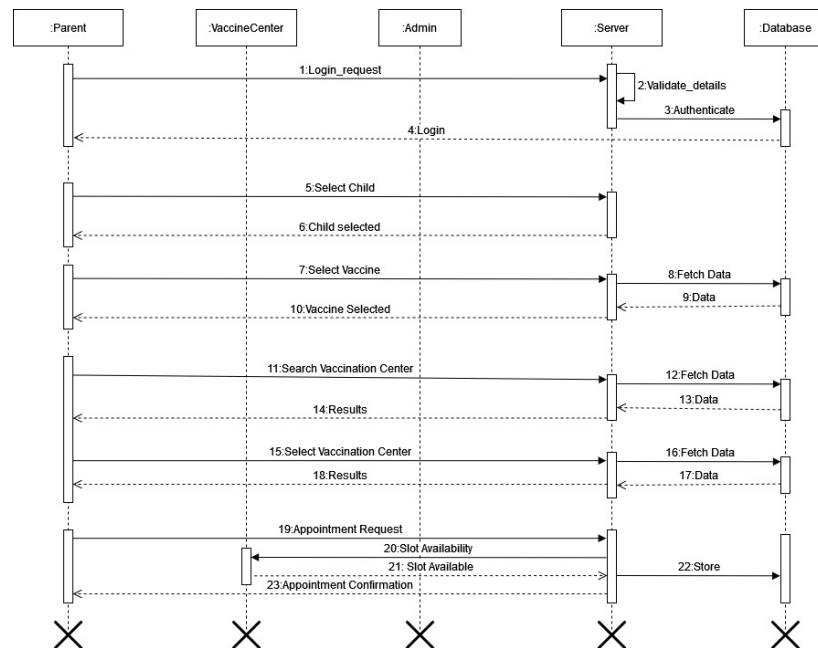


3.5 Sequence Diagram

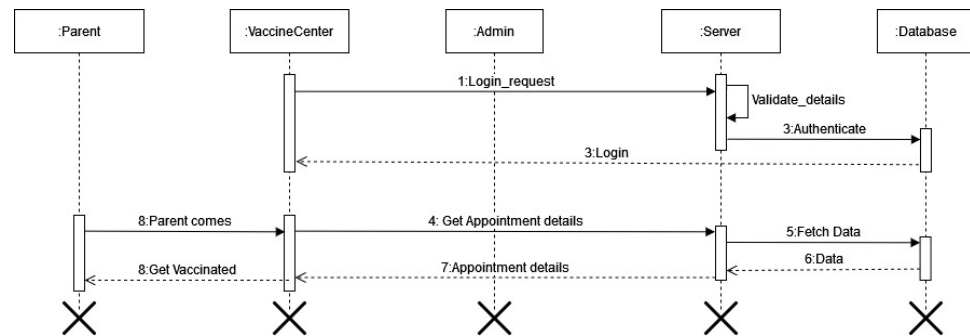
Registration Sequence Diagram:



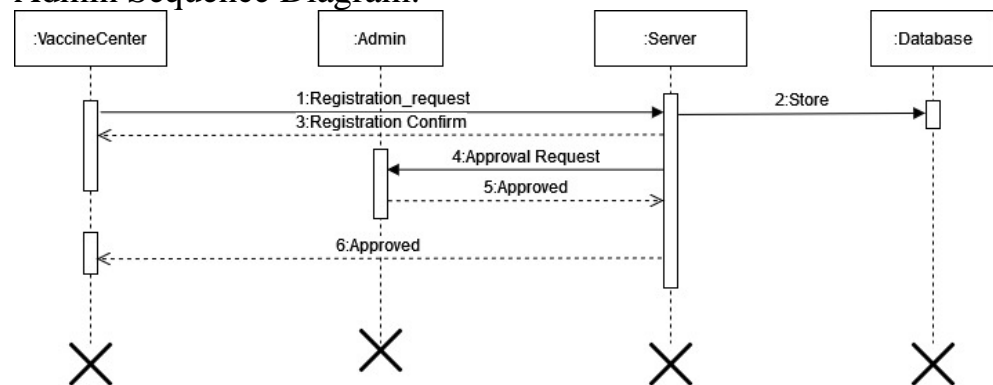
Parent Sequence Diagram:



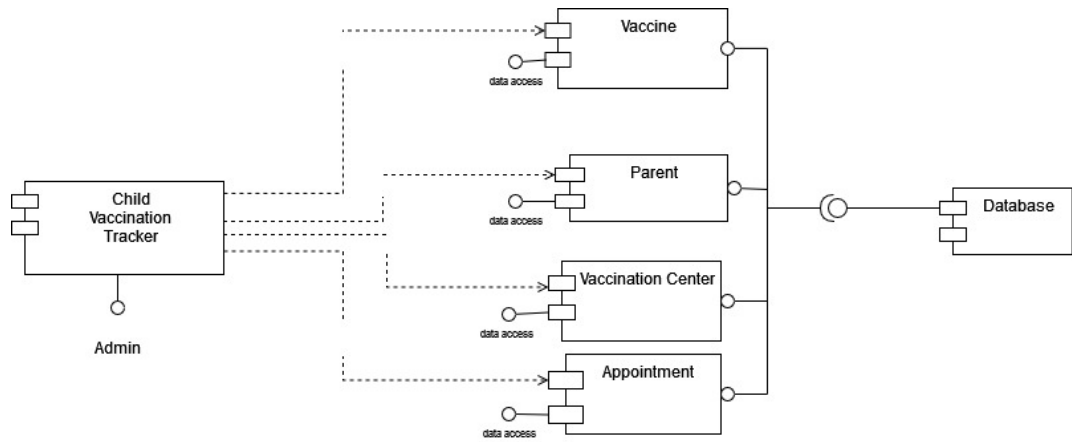
Vaccination Center Diagram:



Admin Sequence Diagram:

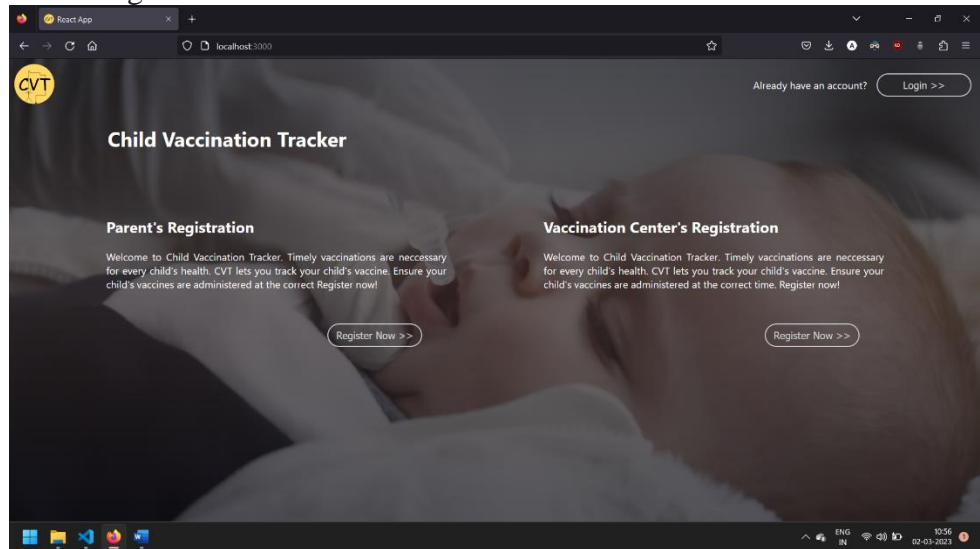


3.6 Component Diagram

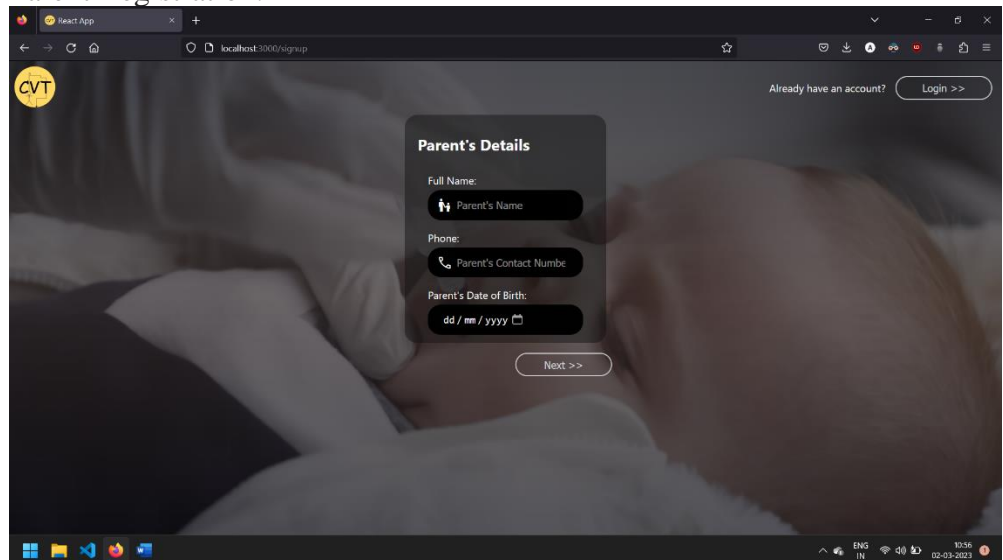


3.7 Sample Input and Output Screens:

Home Page



Parent Registration:



Address Registration:

The screenshot shows a web browser window with the URL `localhost:3000/signup`. The page features a CVT logo in the top left and a "Login >>" button in the top right. The main content is a dark-themed modal titled "Enter your Address". This modal contains nine input fields arranged in a 3x3 grid, each with a corresponding icon: "House No." (house icon), "Locality" (location pin icon), "Road" (road icon), "City/Town/Village" (city icon), "Taluka/Tehsil" (house icon), "District" (map icon), "State" (state outline icon), "Country" (flag icon), and "Pin Code" (pin icon). The "Country" field is pre-filled with "India". At the bottom of the modal are two buttons: "<< Back" and "Next >>". The browser's taskbar at the bottom shows the Windows logo, several application icons, and system information including "ENG IN", signal strength, and the date/time "11:01 02-03-2023".

Child Creation:

The screenshot shows the same web browser window as above, but the modal is titled "Child's Details". It contains three input fields: "Name:" with a "Child's Name" placeholder, "Gender:" with radio buttons for "Male" (selected) and "Female", and "Child's Date of Birth:" with a date picker showing "dd / mm / yyyy". At the bottom of the modal are two buttons: "<< Back" and "Next >>". The browser's taskbar at the bottom shows the same system information as the previous screenshot, with the date/time now showing "11:02 02-03-2023".

Password Creation:

React App

localhost:3000/signup

CVT

Already have an account? [Login >>](#)

Password Creation

Password

Confirm Password

Set a Strong Password.

- Minimum 8 characters
- Minimum 1 uppercase letter
- Minimum 1 lowercase letter
- Minimum 1 special character

<< Back Next >>

Email:

React App

localhost:3000/signup

CVT

Already have an account? [Login >>](#)

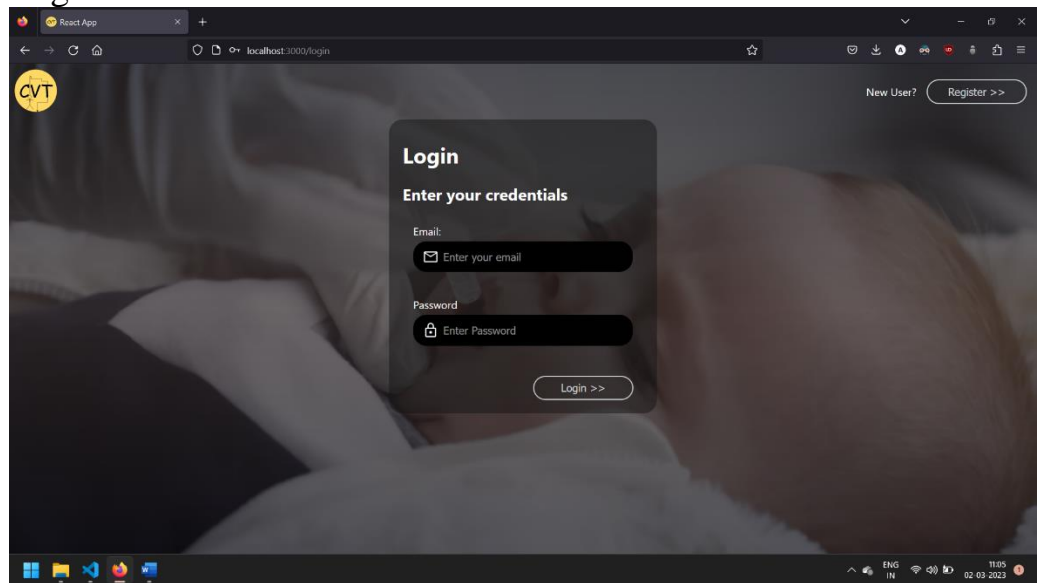
Email

Email:

Confirm Email:

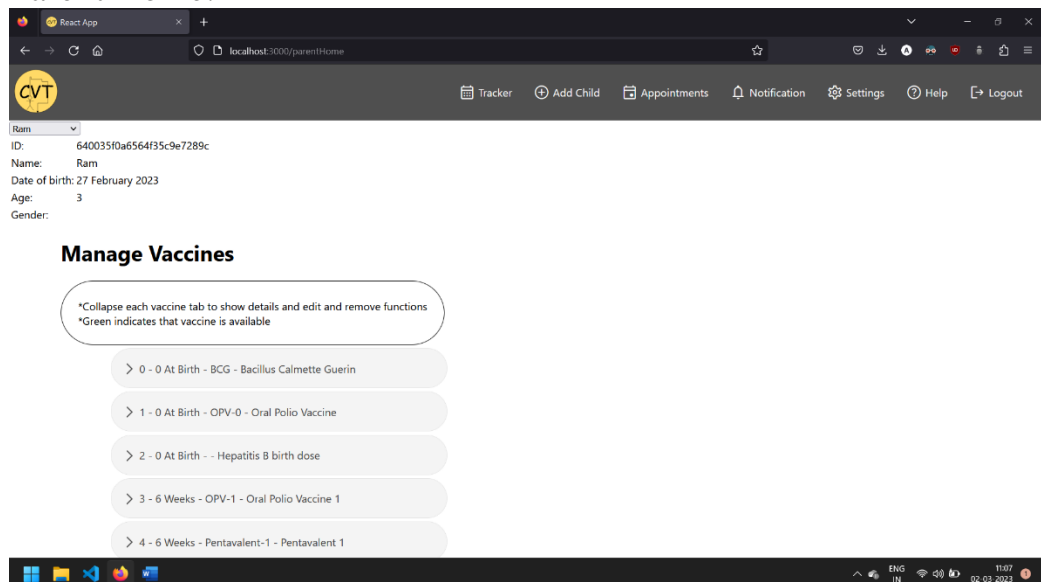
<< Back Sign Up >>

Login:



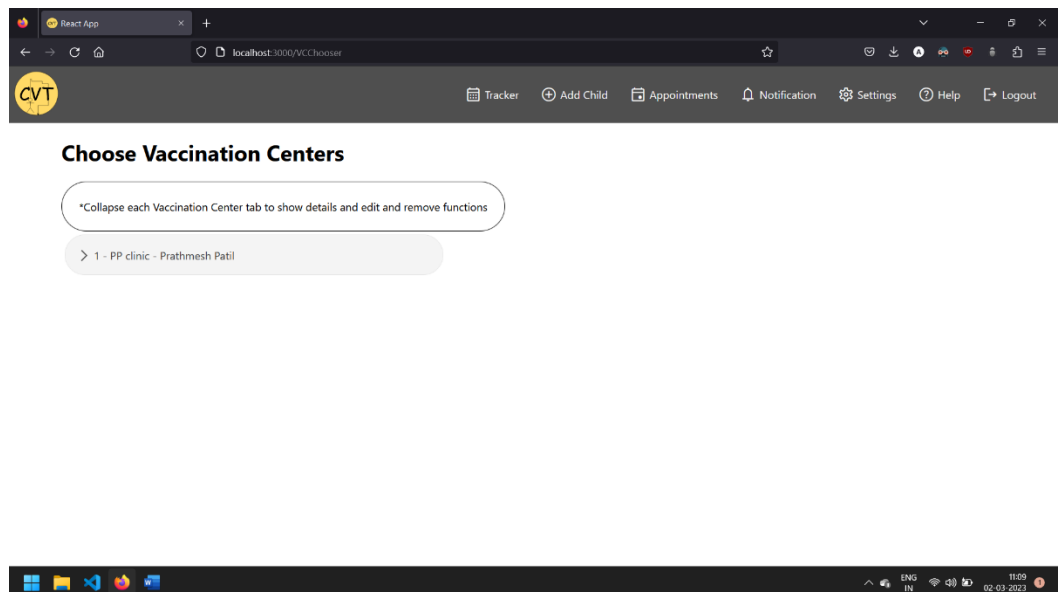
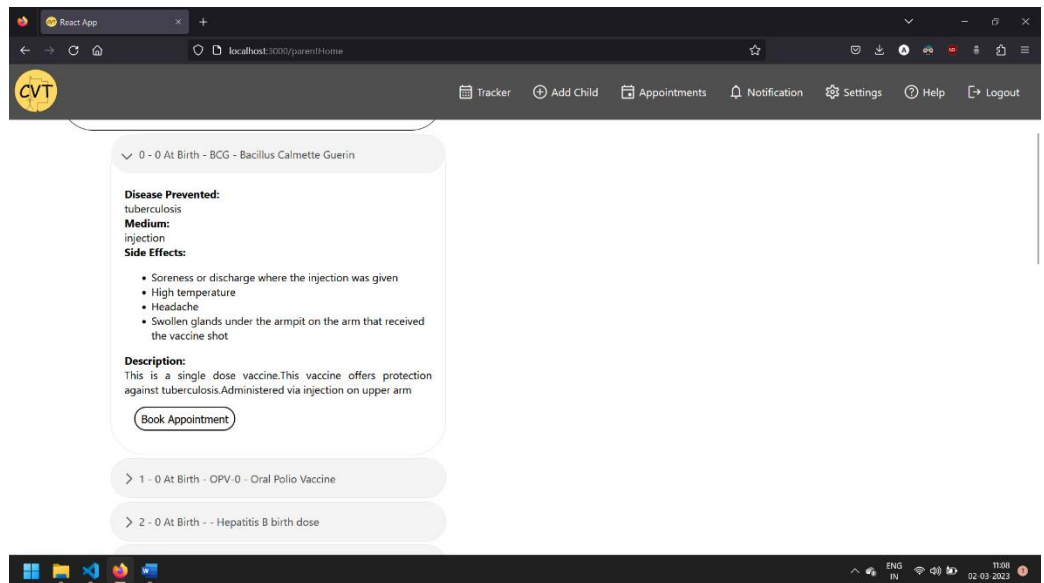
The screenshot shows a web browser window with the URL `localhost:3000/login`. The page features a dark background with a faint image of a baby. In the top left corner is the CVT logo. In the top right corner, there is a link for "New User?" and a "Register >>" button. A central modal box titled "Login" contains the heading "Enter your credentials". Below this, there are two input fields: "Email:" with a placeholder "Enter your email" and "Password:" with a placeholder "Enter Password". At the bottom of the modal is a "Login >>" button. The browser's taskbar at the bottom shows the Windows logo, several application icons, and system status icons including language (ENG IN), network, and time (11:05, 02.03.2023).

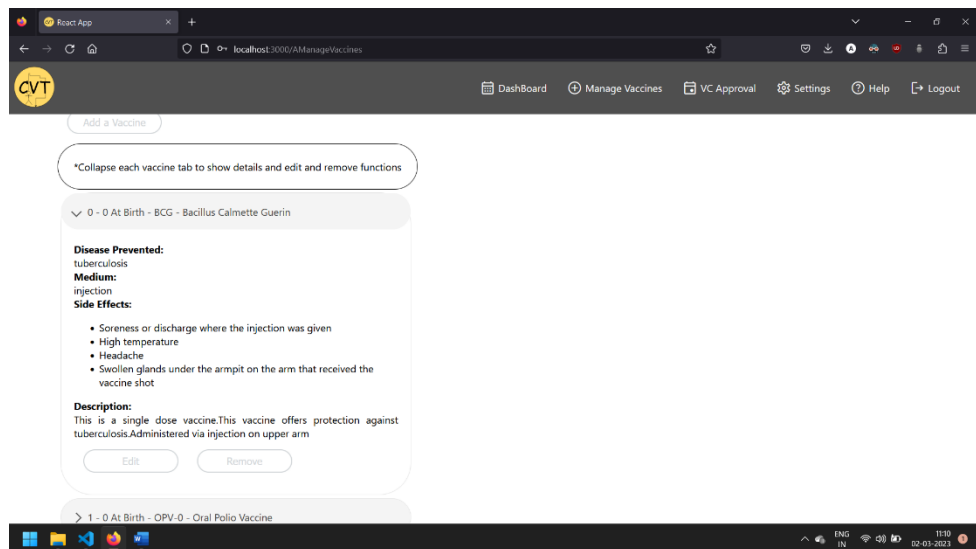
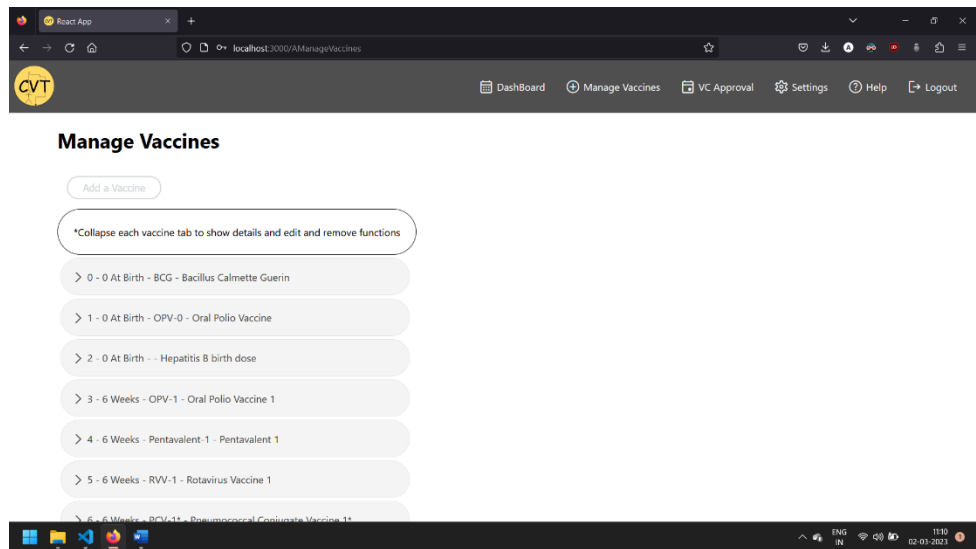
Parent Home:



The screenshot shows a web browser window with the URL `localhost:3000/parentHome`. The page has a dark header with the CVT logo on the left and a navigation menu on the right containing: Tracker, Add Child, Appointments, Notification, Settings, Help, and Logout. Below the header, a dropdown menu is set to "Ram". The user details section displays: ID: 640035f0a6564f35c9e7289c, Name: Ram, Date of birth: 27 February 2023, Age: 3, and Gender: (blank). The main section is titled "Manage Vaccines" and includes two instructions: "*Collapse each vaccine tab to show details and edit and remove functions" and "*Green indicates that vaccine is available". Below these are six vaccine tabs, each with a right-pointing chevron: "0 - 0 At Birth - BCG - Bacillus Calmette Guerin", "1 - 0 At Birth - OPV-0 - Oral Polio Vaccine", "2 - 0 At Birth - - Hepatitis B birth dose", "3 - 6 Weeks - OPV-1 - Oral Polio Vaccine 1", and "4 - 6 Weeks - Pentavalent-1 - Pentavalent 1". The browser's taskbar at the bottom shows the Windows logo, application icons, and system status icons including language (ENG IN), network, and time (11:07, 02.03.2023).

Parent Vaccination:





Vaccine Info Editor:

The screenshot shows a web browser window with the address bar displaying 'localhost:3000/VaccineEditor'. The application has a dark theme and a top navigation bar with a 'CVT' logo and links for 'Dashboard', 'Manage Vaccines', 'VC Approval', 'Settings', 'Help', and 'Logout'. The main content area is titled 'Vaccine Details' and contains several input fields for vaccine information. The background of the form is a faint image of a person's arm.

Vaccine Details

Vaccine Name: Bacillus Calmette Guerir

Vaccine Name: BCG

Start Range: 0 At Birth

End Range: 0 At Birth

Disease Prevented: tuberculosis

Medium: injection

Vaccine Side Effects: Soreness or discharge where the injection was given, High temperature, Headache, Swollen glands under the armpit on the arm that received the vaccine shot

Vaccine Description: This is a single dose vaccine. This vaccine offers protection against tuberculosis. Administered via injection on upper arm

CHAPTER 4: CODING Sample code

```
import bcrypt from "bcrypt";
import User from "../models/User.js";
import Admin from "../models/Admin.js";
import VaccineCenter from "../models/VaccineCenter.js";
import Vaccine from "../models/Vaccine.js";
import { validationResult } from "express-validator";

export const parentRegister = async (req, res, next) => {
  try {
    const {
      parentName,
      Email,
      parentPhone,
      parentDOB,
      address,
      childName,
      childDOB,
      childGender,
      password,
      userType,
    } = req.body;
    const errors = validationResult(req);

    if (!errors.isEmpty()) {
      res.status(400).json({ error: errors.array() });

      // res.status(400).send("{ error: errors.array() }");
    } else {
      const salt = await bcrypt.genSalt();
      const passwordHash = await bcrypt.hash(password, salt);
      let Vaccines = await Vaccine.find().sort({ vSortVar: "ascending" });
      Vaccines = Vaccines.map((obj) => {
        return {
          ...obj,
          Done: false,
          DoneOn: null,
          DoneAt: null,
          DoneOn: null,
        };
      });
      const newUser = new User({
        parentName,
        Email,
        parentPhone,
        parentDOB,
        address,
```



```

        children: [
            {
                childName: childName,
                childDOB: childDOB,
                childGender: childGender,
                Vaccines: Vaccines,
            },
        ],
        password: passwordHash,
        userType,
    });
    const userExists = await User.findOne({ Email: Email });
    if (userExists !== null) {
        res.status(400).json({
            error: [
                {
                    msg: "Email is already registered",
                },
            ],
        });
    } else {
        newUser.save().then(res.status(201).send({ message: "OK" }));
    }
} catch (err) {
    res.status(500).json({ error: err.message });
}
};
// _____

export const vcRegister = async (req, res, next) => {
    try {
        const { vcName, Email, vcPhone, vcOwnerName, address, userType, password }
        =
            req.body;
        const errors = validationResult(req);

        if (!errors.isEmpty()) {
            res.status(400).json({ error: errors.array() });

            // res.status(400).send("{ error: errors.array() }");
        } else {
            const salt = await bcrypt.genSalt();
            const passwordHash = await bcrypt.hash(password, salt);
            const newUser = new VaccineCenter({
                vcName,
                Email,
                vcPhone,
            });
        }
    }
};

```

```

        vcOwnerName,
        address,
        userType,

        password: passwordHash,
    });
    const userExists = await User.findOne({ Email: Email });
    if (userExists !== null) {
        res.status(400).json({
            error: [
                {
                    msg: "Email is already registered",
                },
            ],
        });
    } else {
        newUser.save().then(res.status(201).send({ message: "OK" }));
    }
} catch (err) {
    res.status(500).json({ error: err.message });
}
};
// _____

```

```

export const adminRegister = async (req, res, next) => {
    try {
        const {
            adminName,
            Email,
            adminPhone,
            adminDOB,
            userType,
            level,
            isApproved,
            password,
        } = req.body;
        const errors = validationResult(req);

        if (!errors.isEmpty()) {
            res.status(400).json({ error: errors.array() });

            // res.status(400).send("{ error: errors.array() }");
        } else {
            const salt = await bcrypt.genSalt();
            const passwordHash = await bcrypt.hash(password, salt);
            const newUser = new Admin({
                adminName,

```

```

    Email,
    adminPhone,
    adminDOB,
    userType,
    level,
    isApproved,

    password: passwordHash,
  });
  const userExists = await User.findOne({ Email: Email });
  if (userExists !== null) {
    res.status(400).json({
      error: [
        {
          msg: "Email is already registered",
        },
      ],
    });
  } else {
    newUser.save().then(res.status(201).send({ message: "OK" }));
  }
}
} catch (err) {
  res.status(500).json({ error: err.message });
}
};

export const login = async (req, res) => {
  try {
    const { Email, password } = req.body;
    const errors = validationResult(req);

    if (!errors.isEmpty()) {
      res.status(400).json({ error: errors.array() });
    } else {
      User.findOne({ Email })
        .then((user) => {
          bcrypt
            .compare(password, user.password)
            .then((passwordMatched) => {
              if (!passwordMatched) {
                return res
                  .status(400)
                  .json({ error: [{ msg: "Email/Password Invalid" }] });
              }
              const token = user.GenerateAuthToken();
              res.cookie("token", token);
              console.log({ message: token });
            });
        })
    }
  }
};

```

```

        console.log(user.userType);

        return res
            .status(201)
            .json({ message: "OK", userType: user.userType });
    })
    .catch((error) => {
        return res.status(400).json({
            error: [
                { msg: "Email/Password might be incorrect", er: error },
            ],
        });
    });
});

}
} catch (err) {
    res.status(500).json({ error: err.message });
}
};
// _____

export const addchild = async (req, res) => {
    try {
        const { Email, childName, childDOB, childGender } = req.body;
        const errors = validationResult(req);

        if (!errors.isEmpty()) {
            res.status(400).json({ error: errors.array() });

            // res.status(400).send("{ error: errors.array() }");
        } else {
            let Vaccines = await Vaccine.find()
                .sort({ vSortVar: "ascending" })
                .lean();
            Vaccines = Vaccines.map((obj) => {
                return { ...obj, Done: false, DoneOn: "", DoneAt: "", DoneOn: "" };
            });
            console.log(Vaccines);
            const user = await User.findOne({ Email: Email });
            const child = {
                childDOB: childDOB,
                childGender: childGender,
            }

```

```

        childName: childName,
        Vaccines: Vaccines,
    };
    //console.log(user.children);
    user.addChild(child).then(res.status(201).send({ message: "OK" }));
}
} catch (err) {
    res.status(500).json({ error: err.message });
}
};
// _____

```

```

export const checkEmail = async (req, res) => {
    try {
        const { Email } = req.body;

        const userExists = await User.findOne({ Email: Email });
        if (userExists != null) {
            res.status(400).json({
                error: [
                    {
                        msg: "Email is already registered",
                    },
                ],
            });
        } else {
            res.status(201).send({ message: "OK" });
        }
    } catch (err) {
        res.status(500).json({ error: err.message });
    }
};
// _____

```

```

export const logout = async (req, res) => {
    // res.cookie("token", null)
    // res.clearCookie('token');
    res.clearCookie("token");
    return res.status(201).send({ message: "OK" });
};
// _____

```

```

export const adminGetVC = async (req, res, next) => {
    try {
        const errors = validationResult(req);

        if (!errors.isEmpty()) {
            return res.status(400).json({ error: errors.array() });
        }
    }
}

```

```

    const VC = await User.find({ userType: { $in: ["v", "vr", "vrr", "vu"] }
});
    console.log(VC);

    if (VC.length > 0) {
        return res.json({ VC: VC });
    } else {
        return res.send({ VC: [] });
    }
} catch (err) {
    res.status(500).json({ error: err.message });
}
};

export const userGetVC = async (req, res, next) => {
    try {
        const { vid } = req.body;
        console.log(vid);
        const errors = validationResult(req);

        if (!errors.isEmpty()) {
            return res.status(400).json({ error: errors.array() });
        }

        // const VC = await User.find({
        //     userType: { $in: ["v"] },
        //     vaccines: { $elemMatch: { _id: vid } }
        // });
        const VC = await User.find({ userType: 'v', vaccines: { $in: [vid] } });
        console.log("_____", VC);

        if (VC.length > 0) {
            return res.json({ VC: VC });
        } else {
            return res.json({ VC: [] });
        }
    } catch (err) {
        res.status(500).json({ error: err.message });
    }
};

// _____
export const adminGetVaccines = async (req, res, next) => {
    try {
        const errors = validationResult(req);

        if (!errors.isEmpty()) {
            res.status(400).json({ error: errors.array() });
        }
    }
};

```

```

    // res.status(400).send("{ error: errors.array() }");
  } else {
    const Vaccines = await Vaccine.find().sort({ vSortVar: "ascending" });
    if (Vaccines !== null) {
      res.status(200).json({ Vaccines: Vaccines });
    } else {
      res.status(200).send({ Vaccines: [] });
    }
  }
} catch (err) {
  res.status(500).json({ error: err.message });
}
};
// _____
export const addVaccine = async (req, res) => {
  try {
    const {
      _id,
      vName,
      vShortName,
      vSideEffects,
      vStartRangeNumber,
      vStartRangePost,
      vEndRangeNumber,
      vEndRangePost,
      medium,
      vDiseasePrevented,
      vDesc,
      vSortVar,
      mode,
    } = req.body;
    const errors = validationResult(req);

    if (!errors.isEmpty()) {
      res.status(400).json({ error: errors.array() });

      // res.status(400).send("{ error: errors.array() }");
    } else {
      // if mode is update then execute updatequery
      // if mode is add then execute save
      if (mode === "add") {
        console.log("adding");
        const vaccine = await Vaccine.findOne({ vName: vName });
        if (vaccine !== null) {
          return res
            .status(400)
            .json({ error: vName + " vaccine already exists" });
        } else {

```

```

    const vaccine = new Vaccine({
      vName,
      vShortName,
      vSideEffects,
      vStartRangeNumber,
      vStartRangePost,
      vEndRangeNumber,
      vEndRangePost,
      medium,
      vDiseasePrevented,
      vDesc,
      vSortVar,
    });

    vaccine.save().then(res.status(201).send({ message: "OK" }));
  }
} else if (mode == "update") {
  console.log("updating", _id);

  const vaccine = {
    vName,
    vShortName,
    vSideEffects,
    vStartRangeNumber,
    vStartRangePost,
    vEndRangeNumber,
    vEndRangePost,
    medium,
    vDiseasePrevented,
    vDesc,
    vSortVar,
  };
  console.log(vaccine);

  Vaccine.findOneAndUpdate(
    { _id },
    vaccine,
    { new: true },
    (error, document) => {
      if (error) {
        console.log(error);
      } else {
        res.status(201).json(document);
      }
    }
  );
}
}
}

```



```

    } catch (err) {
      res.status(500).json({ error: err.message });
    }
  };
  //

```

```

export const removeVaccine = async (req, res) => {
  try {
    const { _id } = req.body;
    const errors = validationResult(req);

    if (!errors.isEmpty()) {
      res.status(400).json({ error: errors.array() });
    } else {
      Vaccine.findOneAndDelete({ _id: _id }, (error, document) => {
        if (error) {
          console.log(error);
        } else {
          res.status(201).json(document);
        }
      });
    }
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
};
//

```

```

export const ManageVC = async (req, res) => {
  try {
    const { _id, userType } = req.body;
    const errors = validationResult(req);

    if (!errors.isEmpty()) {
      res.status(400).json({ error: errors.array() });
    } else {
      {
        User.findOneAndUpdate(
          { _id },
          {
            userType: userType,
          },
          { new: true },
          (error, document) => {
            if (error) {
              console.log(error);
            } else {
              res.status(201).json(document);
            }
          }
        );
      }
    }
  }
};

```

```

        }
      }
    );
  }
}
} catch (err) {
  res.status(500).json({ error: err.message });
}
};
// _____

export const removeVCVaccine = async (req, res) => {
  try {
    const { _id, Email } = req.body;
    const errors = validationResult(req);

    if (!errors.isEmpty()) {
      res.status(400).json({ error: errors.array() });
    } else {
      console.log(Email);
      await VaccineCenter.updateOne(
        { Email: Email },
        { $pull: { vaccines: _id } }
      );
      return res.status(200).send({ message: "OK" });
    }
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
};
// _____

export const addVCVaccine = async (req, res) => {
  try {
    const { _id, Email } = req.body;
    const errors = validationResult(req);
    console.log(_id, Email);
    if (!errors.isEmpty()) {
      return res.status(400).json({ error: errors.array() });
    } else {
      await VaccineCenter.updateOne(
        { Email: Email },
        { $push: { vaccines: _id } }
      );
      return res.status(200).send({ message: "Vaccine added successfully." });
    }
  } catch (err) {
    return res.status(500).json({ error: err.message });
  }
};

```

```
}  
};  
// _____
```

```
body {  
  margin: 0;  
  padding: 0;  
  border: none;  
}  
  
/* color pallet */  
:root {  
  --background: #161621;  
  --primary: #ffffff;  
  --secondary: #ffffff;  
  --accent: #4f4f4f;  
}  
.background{  
  background-color: var(--background);  
}  
.primary {  
  color: var(--primary);  
}  
  
.secondary {  
  color: var(--secondary);  
}  
  
.bAccent {  
  background-color: var(--accent);  
}  
  
.fAccent {  
  color: var(--accent);  
}  
  
.fWhite {  
  color: white !important;  
}  
.fRed{color: red;}  
.fGreen{color: green;}
```

```
.bGreen{background-color: green !important;}
/* Font sizing */
.font-1 {
    font-size: 1em;
}

.font-1dot5 {
    font-size: 1.5em;
}

.font-2 {
    font-size: 2rem;
}

.font-4 {
    font-size: 4rem;
}

.font-6 {
    font-size: 6rem;
}

.font-8 {
    font-size: 8rem;
}

.font-10 {
    font-size: 10rem;
}

.font-12 {
    font-size: 12rem;
}

.font-14 {
    font-size: 14rem;
}

.font-16 {
    font-size: 16rem;
}

.font-18 {
    font-size: 18rem;
}

.font-20 {
    font-size: 20rem;
}
```

```

}

.font-bold {
  font-weight: bold;
}

.font-title {
  font-family: 'Lucida Sans', 'Lucida Sans Regular', 'Lucida Grande', 'Lucida
Sans Unicode', Geneva, Verdana, sans-serif;
}

.font-subtitle {
  font-family: 'Source Sans Pro', sans-serif;
;
}

.uppercase {
  text-transform: uppercase;
}

.lowercase {
  text-transform: lowercase;
}

.font-center {
  text-align: center;
}

.v-align {
  position: fixed;
  top: 45%;
  left: 50%;
  transform: translate(-50%, -50%);
  white-space: nowrap;
}

/* .text-dark-gradient{
  -webkit-mask-image: -webkit-gradient(linear, left top,
    left bottom, from(rgb(166, 166, 166)), to(rgba(0, 0, 0, 0)));
} */
/* Logo */
.logo-small {
  width: 4em;
}.rmargin-10{
  margin-right: 10px;
}

```

```

.centered{
  display: flex;
  align-items: center;
}

/* Not found page */

.NotFound-container {
  background-image:
    linear-gradient(rgba(0, 0, 0, 0.55),
      rgba(0, 0, 0, 0.60)), url(../images/404bg.gif);
  background-repeat: repeat;
  height: 100vh;
  background-position: center;
  background-size: 150px;
  background-color: rgb(252, 254, 252);
}

/* General */

input::-webkit-inner-spin-button {
  -webkit-appearance: none;
  margin: 0;
}

/* Firefox */
input[type="number"] {
  -moz-appearance: textfield;
}

.shape {
  width: 0px;
}

/* Loading Page */
.loading-container{
  overflow: hidden;
}
.loading-animation{
  width: 500px;
  height: 500px;
  position: absolute;
  bottom: 20%;
}

```

```

    animation: linear infinite;
    animation-name: run;
    animation-duration: 3s;
}
.loading-animation img{
    width: 50%;
}

.loading-text{
    position: absolute;
    font-family: 'Courier New', Courier, monospace;
    font-size: 2em;
    bottom: 50%;
    left: 0;
    width: 100%;
    text-align: center;
}

/* Animations */
@keyframes run {
    0% {
        right: -20%;
    }

    100% {
        right: 100%;
    }
}

.zoom-in-zoom-out1 {
    animation: zoom-in-zoom-out1 10s ease-out infinite;
}

@keyframes zoom-in-zoom-out1 {
    0% {
        transform: scale(1, 1);
    }

    50% {
        transform: scale(1.5, 1.5);
    }

    100% {
        transform: scale(1, 1);
    }
}

```

```

.zoom-in-zoom-out2 {
  animation: zoom-in-zoom-out2 5s ease-out infinite;
}

@keyframes zoom-in-zoom-out2 {
  0% {
    transform: scale(1, 1);
  }

  50% {
    transform: scale(1.5, 1.5);
  }

  100% {
    transform: scale(1, 1);
  }
}

/* Non Login Background */
.auth-background {
  background:
    linear-gradient(rgba(15, 14, 20, 0.70),
      rgba(50, 41, 40, 0.70),
      rgba(15, 14, 20, 0.70)),
    url(../images/child.png);
  background-repeat: no-repeat;
  height: 100vh;
  background-position: center;
  background-size: cover;
  background-attachment: local;
  overflow: scroll;
}

/* Home */
.home-container {

  display: flex;
  justify-content: flex-start;
  flex-direction: column;

}

.empty-col {
  display: flex;
  height: 20vh;
  width: 100vw;
  justify-content: space-evenly;
}

```



```

}

.empty-row {
  display: flex;
  height: 100vh;
  width: 10vw;
}

.empty-row-by-2 {
  display: flex;

  width: 45%;
}

.home-division {

  z-index: 10;
  display: flex;
  justify-content: space-evenly;
  text-align: justify;

}

.home-division .right {
  display: flex;

  width: 35%;

  align-items: center;
}

.home-division .left {
  display: flex;
  width: 35%;
  align-items: center;
}

.top {
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.top .top-item {
  display: flex;

```

```

    align-items: center;
}

.top .logo {
    display: flex;
    align-self: flex-start;
    margin: 0.5rem;
}

.home-division .left .btn {
    display: flex;
    margin-left: 65%;
}

.home-division .right .btn {
    display: flex;
    margin-left: 65%;
}

.btn {
    padding: 7px;
    width: 150px;
    margin: 15px;
    white-space: normal;
    word-wrap: break-word;
    border-radius: 20px;

    background-color: transparent;
    border: 2px solid #D2D6D9;
    color: #D2D6D9;
    font-size: 1em;
}

.btn2 {
    padding: 7px;
    width: 150px;
    margin: 15px;
    white-space: normal;
    word-wrap: break-word;
    border-radius: 20px;

    background-color: transparent;
    border: 2px solid #000000;
    color: #000000;
    font-size: 1em;
}

.home-container .btn:hover {

```

```

    cursor: pointer;
}

/* forminput */
.material-symbols-outlined {
    align-self: center;

    font-variation-settings:
        'FILL' 0,
        'wght' 400,
        'GRAD' 0,
        'opsz' 48
}

.formparts-container {
    display: flex;
    justify-content: space-around;
    flex-direction: row;
}

.signup-container .left {
    display: flex;
    justify-self: flex-start;
    flex-direction: column;
}

.signup-container .right {
    display: flex;
    justify-self: flex-end;
    flex-direction: column;
}

/* .formicon{
    background-color: white;
}
*/

.iconInputContainer {
    display: flex;
    flex-direction: row;

    background-color: rgb(1, 1, 1);
    border-radius: 20px;
    padding: 7px 25px 7px 14px;
    margin: 0.3em 0 0.5em 0;
}

```

```

.Form-input {
  border: none;
  width: min-content;
  background-color: black;
  height: 30px;
  margin-left: 0.3em;
  color: white;
  outline: none;
  font-weight: normal;
  font-size: 16px;
}

```

```

.regform {
  display: flex;
  justify-content: center;
  align-items: center;
  width: 100%;
  flex-direction: column;
  position: relative;
  top: 0;
  bottom: 0;
  left: 0;
  right: 0;

  margin: auto;
}

```

```

.reg-container {
  display: flex;
  justify-content: space-between;
  flex-direction: column;
}
.flex-row{
  display: flex;
  flex-direction: row;
}
.flex-item{
  padding: 0.5rem 1rem 0 1rem;
}
input[type="radio"]{
  background-color: black;
}

```

```

.RegStep{
  background-color: rgba(20, 20, 20, 0.5);
  padding: 10px 20px 5px 20px;

  border-radius:20px ;
}
.btn:disabled{
visibility: hidden;
border: none;
color: gray;
background-color: rgba(0, 0, 0, 0.7);
}
/* custom radio buttons */

.tbtn {

  display: inline-block;
  padding:5px 10px;
  border-radius: 20px;
  border: 1px solid transparent;
  transition: background-color 100ms step-end, color 100ms ease;
}
.invisible-radio{
  display: none;
}
.radio-container{
  margin-top: 10px;
  border: 3px solid transparent;
  border-radius: 20px;
  background-color: black;
  color: #898989;
  width:max-content;
}
input[type="radio"].toggle:checked+label{
  background-color: rgb(255, 255, 255);
  color: #000000;
  border: 1px solid #cacaca;
  transition: border 100ms ease;
}
.reg-step{

box-shadow: -2px 18px 24px 0px rgba(255, 255, 255, 0.75);
}

/* Navbar*/

```

```

.nav-image-items {
  display: flex;
  width: 100vw;
  justify-content: space-around;
}
.nav-bar{
margin: 0;

display: flex;
}
.nav-links {
  display: flex;
  width: 100%;
  justify-content:right;

  margin-right: 1em;
}

.nav-item {
  display: flex;

  padding: 1em;
  align-items: center;
}

}
.nav-item:hover {
  background-color: var(--secondary);
  color: var(--background);
}

.nav-items:focus {
  background-color: var(--background);
}
.nav-icon{
  margin-right: 0.2em;
}
.navContainer{
  height: 10%;
  position: sticky;
  top: 0px;
  width: 100%;
}

}

/* ParentHome */
.belowNav{

```

```

padding-top: 0%;

}

.checked_condition{
    color: green;
}
.unchecked_condition{
    color: red;
}

.navContainer{
    z-index: 10;
}
.belowNav{
    z-index: 9;
}

/* Accordion */
/**
 * -----
 * Demo styles
 * -----
 */
.text-justify{
    text-align: justify;
}
.Margin5{
    margin:auto 5%;
}
.accordion {
    border: 1px solid rgba(0, 0, 0, 0.1);
    border-radius: 50px;
    margin: 5px;
    width: 40%;
}

.accordion__item + .accordion__item {
    border-top: 1px solid rgba(0, 0, 0, 0.1);
}

.accordion__button {
    background-color: #f4f4f4;
    color: #444;
    cursor: pointer;
    padding: 18px;

```

```

    border-radius: 50px;

    text-align: left;
    border: none;
}

.accordion__button:hover {
    background-color: #ddd;
}

.accordion__button:before {
    display: inline-block;
    content: '';
    height: 10px;
    width: 10px;
    margin-right: 12px;
    border-bottom: 2px solid currentColor;
    border-right: 2px solid currentColor;
    transform: rotate(-45deg);
}

.accordion__button[aria-expanded='true']::before,
.accordion__button[aria-selected='true']::before {
    transform: rotate(45deg);
}

[hidden] {
    display: none;
}

.accordion__panel {
    padding: 20px;
    animation: fadein 0.35s ease-in;
}

/* ----- */
/* ----- Animation part ----- */
/* ----- */

@keyframes fadein {
    0% {
        opacity: 0;
    }

    100% {
        opacity: 1;
    }
}

```



```

.absolute-right{
display: flex;
justify-content: center;
width: 30%;
align-self: center;
margin:auto;
flex-direction: column;

}

.Note{
width: fit-content;
padding: 25px;

border-radius: 50px;
border: 1px solid #000000;
}

import { useState, useEffect, useContext } from "react";

import NavBar from "../components/NavBar";
import TrackerItem from "../components/TrackItem";
import { ToastContainer } from "react-toastify";
import "react-toastify/dist/ReactToastify.css";
import { useNavigate } from "react-router-dom";
import Loading from "../components/Loading";
import { notifyToast } from "../functions/notifyToast";
import { UserContext } from "../App";
import {
  Accordion,
  AccordionItem,
  AccordionItemHeading,
  AccordionItemButton,
  AccordionItemPanel,
} from "react-accessible-accordion";
function ParentHome() {
  let { state, dispatch } = useContext(UserContext);

  const getAge = (dob1) => {
    var today = new Date();
    var birthDate = new Date(dob1); // create a date object directly from
`dob1` argument
    var age_now = today.getFullYear() - birthDate.getFullYear();
    var m = today.getMonth() - birthDate.getMonth();
    const diffMilliseconds = Math.abs(today - birthDate);
    const days = Math.floor(diffMilliseconds / 86400000);
  }

```

```

const weeks = Math.floor(diffMilliseconds / 86400000 / 7);
// console.log(( Date.parse(today)-Date.parse(birthDate))/86400 )
//var w=Math.floor((Date.parse(today)-Date.parse(birthDate) )/604800);
if (m < 0 || (m === 0 && today.getDate() < birthDate.getDate())) {
  age_now--;
}
return days;
};
const readableDate = (inputDate) => {
  const date = new Date(inputDate);
  const options = {
    year: "numeric",
    month: "long",
    day: "numeric",
    timeZone: "UTC",
  };
  return date.toLocaleDateString(undefined, options);
};
const navigate = useNavigate();

const [selectValue, setSelectValue] = useState("");

function handleSelectChange(event) {
  setSelectValue(event.target.value);
  console.log(selectValue);
}
const [userData, setUserData] = useState();
const [loading, setLoading] = useState(true);
const searchChild = (arr, id) => {
  for (const ele of arr) {
    if (ele._id === id) {
      console.log(ele);
      return ele;
    }
  }
};
const removeVCVaccine = () => 0;
const addVCVaccine = () => 0;
const pageParentHome = () => {
  fetch("/ParentHome", {
    method: "get",
    headers: {
      Accept: "*/*",
      "Content-Type": "application/json",
    },
    credentials: "include",
  }).then((res) => {
    if (res.status === 201) {

```

```

        res.json().then((json) => {
            setUserData(json);
            localStorage.setItem("Email", json.Email);
            setLoading(false);
            dispatch({ type: "USER", payload: false });
        });
    }
    if (res.status === 401) {
        navigate("/login");
    }

    // response .then(res => res.json())
    // .then(json => console.log(json))
});
};

useEffect(() => {
    const userType = localStorage.getItem("userType");
    if (userType !== "p") {
        switch (userType) {
            case "v":
                notifyToast("Unauthorised", "error");
                navigate("/vcdash");
                break;
            case "vr":
                notifyToast("Unauthorised", "error");
                navigate("/vcdash");
                break;
            case "vrr":
                notifyToast("Unauthorised", "error");
                navigate("/vcdash");
                break;
            case "vu":
                notifyToast("Unauthorised", "error");
                navigate("/vcdash");
                break;
            case "a":
                notifyToast("Unauthorised", "error");
                navigate("/Admin");
                break;
            default:
                notifyToast("Login First", "error");
                localStorage.setItem("userType", "UNKNOWN");
                navigate("/login");
                break;
        }
    }
}
);

```

```

    pageParentHome();
  }, []);

  if (loading) {
    return <Loading />;
  }
  const x = searchChild(userData.children, selectValue) ||
  userData.children[0];
  return (
    <div>
      <div className="Parentcontainer">
        {/* <NavBar className="topnavbar" /> */}

        <div className="belowNav">
          <select
            onChange={handleSelectChange}
            defaultValue={userData.children[0]}
            placeholder="Select Child"
          >
            {userData.children.map((child) => {
              return (
                <option key={child._id} value={child._id}>
                  {child.childName}
                </option>
              );
            })}
          </select>

          <table className="table">
            <tbody>
              <tr>
                <td>ID:</td>
                <td>{x._id}</td>
              </tr>
              <tr>
                <td>Name:</td>
                <td>{x.childName}</td>
              </tr>

              <tr>
                <td>Date of birth:</td>

                <td> {readableDate(x.childDOB)}</td>
              </tr>

              <tr>
                <td>Age:</td>
                <td>{getAge(x.childDOB)}</td>
              </tr>
            </tbody>
          </table>
        </div>
      </div>
    );
  );
}

```

```

    </tr>

    <tr>
        <td>Gender:</td>
        <td>{x.childGender}</td>
    </tr>
</tbody>
</table>

<div className="tracker">
    <div className="vMgmntActions Margin5">
        <h1>
            Manage Vaccines <br />
        </h1>

        <div className="Note">
            *Collapse each vaccine tab to show details and edit and remove
            functions
            <br />
            *Green indicates that vaccine is available
        </div>

        <div className="vaccine-list Margin5">
            {x.Vaccines.map((item, index) => {
                return (
                    <Accordion key={item._id} allowZeroExpanded>
                        <AccordionItem>
                            <AccordionItemHeading>
                                <AccordionItemButton
                                    className={
                                        item.Done
                                        ? "bGreen fWhite accordion__button"
                                        : "accordion__button"
                                    }
                                >
                                    {index} - {item.vStartRangeNumber}{ " " }
                                    {item.vStartRangePost} - {item.vShortName} -{" " }
                                    {item.vName}
                                </AccordionItemButton>
                            </AccordionItemHeading>
                            <AccordionItemPanel>
                                <b className="v-accordion-key">Disease Prevented:</b>
                                <div className="v-accordion-value">
                                    {item.vDiseasePrevented}
                                </div>
                                <b className="v-accordion-key">Medium:</b>
                                <div className="v-accordion-value">{item.medium}</div>

```

```

        <b className="v-accordion-key">Side Effects:</b>
        <div className="v-accordion-value">
            <ul>
                {item.vSideEffects.map((se) => {
                    return <li key={se}>{se}</li>;
                })}
            </ul>
        </div>
        <b className="v-accordion-key">Description:</b>
        <div className="v-accordion-value text-justify">
            {item.vDesc}
        </div>
        { /* { <button className="button btn"
onClick={()=>{navigate('/VaccineEditor',{state:{item}})}}>Edit</button> */}
        {
            item.Done
            ?(
                <button
                    className="button btn"
                    onClick={() => {
                        removeVCVaccine(item._id);
                    }}
                >
                    Remove
                </button>
            ) : (
                <button
                    className="button btn2"
                    onClick={() => navigate("/VCChooser", {
state: {vid:item._id,pid:userData._id,cid:x._id} })}
                >
                    Book Appointment
                </button>
            )}
        </AccordionItemPanel>
    </AccordionItem>
</Accordion>
    );
    })}
</div>
</div>
</div>
</div>
</div>
);
}

```

```
export default ParentHome;
```

```
import mongoose from "mongoose";
import Jwt from "jsonwebtoken";

const UserSchema = new mongoose.Schema({
  parentName: {
    type: String,
    require: true,
    min: 2,
    max: 50,
  },
  Email: {
    type: String,
    require: true,
    min: 2,
    max: 50,
    unique: true,
  },
  parentPhone: {
    type: Number,
    require: true,
    length: 10,
  },
  parentDOB: {
    type: Date,
    require: true,
  },
  // address: {
  //   type: String,
  //   require: true,
  // },
  address: [
    {
      addrHouseNo: {
        type: String,
        require: true,
      },
      addrLocality: {
        type: String,
        require: true,
      },
    },
  ],
});
```

```

    },
    addrRoad: {
      type: String,
      require: true,
    },
    addrCity: {
      type: String,
      require: true,
    },
    addrTaluka: {
      type: String,
      require: true,
    },
    addrDistrict: {
      type: String,
      require: true,
    },
    addrState: {
      type: String,
      require: true,
    },
    addrCountry: {
      type: String,
      require: true,
    },
    addrPinCode: {
      type: String,
      require: true,
    },
  },
],
children: [
  {
    childName: {
      type: String,
      require: true,
    },
    childDOB: {
      type: Date,
      require: true,
    },
    childGender: {
      type: String,
      require: true,
    },
    Vaccines:[],
  },
],

```



```

password: {
  type: String,
  require: true,
  min: 6,
},
token: {
  type: String,
},
userType: {
  type: String,
},
});

UserSchema.methods.GenerateAuthToken = function () {
  try {
    // const token="TOKEN"
    const token = Jwt.sign({ _id: this._id }, process.env.SECRET, {
      expiresIn: "24H",
    });
    this.token = token;
    this.save();
    return token;
  } catch (err) {
    console.log(err);
  }
};

UserSchema.methods.addChild = async function (child) {
  try {

    this.children = this.children.concat(child);
    await this.save();

  } catch (error) {
    console.log(error);
  }
};

const User = mongoose.model("User", UserSchema);
export default User;

```

CHAPTER 5: LIMITATIONS OF SYSTEM

- In order for the web application to be used it is necessary to have an active internet connection.
- This application requires for vaccination centers to be registered and approved with the admin of the application because of nonexistence of an open centralized government database for the same.

CHAPTER 6: PROPOSED ENHANCEMENTS

- In the future this web application can have a community for the parents to share their stories and experiences and have interactions.
- The users could give ratings and reviews for the vaccination centers which currently are verified and approved by the admin.
- This application can be developed to support for mobile devices such as android and IOS natively through an app.

CHAPTER 7: CONCLUSION

In conclusion, the Child Vaccination Tracker is a valuable tool for parents to ensure that their children receive timely and necessary vaccinations.

With the aid of this web application, parents can keep track of their child's immunization schedule, get timely notifications, access information about the vaccinations, book appointments with registered vaccination centers, and locate them with ease. It is a beneficial resource for vaccination centers to create their profiles, displaying details about their services and location. Ultimately, this application plays a crucial role in promoting the health and well-being of children, by preventing life-threatening diseases that vaccines can prevent.

CHAPTER 8: BIBLIOGRAPHY

- “Know Your Child’s Vaccination Schedule,” *UNICEF India*, n.d., <https://www.unicef.org/india/stories/know-your-childs-vaccination-schedule>.
- “MERN Stack” *mongodb*, n.d., <https://www.mongodb.com/mern-stack>.