Web based Applications for Insurance Services

FINAL REVIEW REPORT

Submitted by

Ekansh Gupta (18BCI0175)

Mihir Srivastava (18BCI0214)

Laksh Gupta (18BCI0190)

Sumeet Roy Kurian (18BCI0188)

Prepared For

DATABASE SYSTEMS (CSE2004) – PROJECT COMPONENT

Submitted To

Dr. Ramanathan L

Assistant Professor (SG)

School of Computer Science and Engineering

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

INDEX

Abstract
Insurance is a means of protection from financial loss. It is a form of risk management, primarily used to hedge against the risk of a contingent or uncertain loss. In these days it is a necessity for everyone to own insurances against different kinds of threats.
So we came up with an idea of providing insurance services to people by using a website. This website caters people of different of sects of society with a variety of insurances. One can select the desired insurance types and premium according to his income and demand. This project uses concepts of database and web development. In this project we are making an insurance website which will take information from user. Our website has a user friendly user interface and it is being created using HTML-CSS and Bootstrap. Our project uses mongo database to store and retrieve data from database. Node JS is used as a tool for connectivity between databases and front-end. This project is a great learning experience especially for the concepts of database and its practical application.

Introduction
Insurance website utilizes concept of web development along with database organization. For frontend web development we have utilized HTML-CSS and Bootstrap. We have created multiple linked webpages. Our homepage contains multiple insurance options for user to choose along with a user login and user sign up options.
Multiple insurance options provided in this webpages utilize concept of hyperlinks to link different web pages. Each webpage connected to home page consist different forms to collect data from user.
The data which is collected here is stored to mongo database. Connecting our mongo database with our web page required utilization of back-end development through Node js.

Requirements
1. **Software Requirements**
    - MONGO DB SHELL
    - NODE JS
    - VS CODE
    - EXPRESS JS
2. **Hardware Requirements**
    - Processor-ryzen3 or above/Intel- i3 3rd gen or above
    - Graphic card- UHD 520
    - RAM 2gb or above

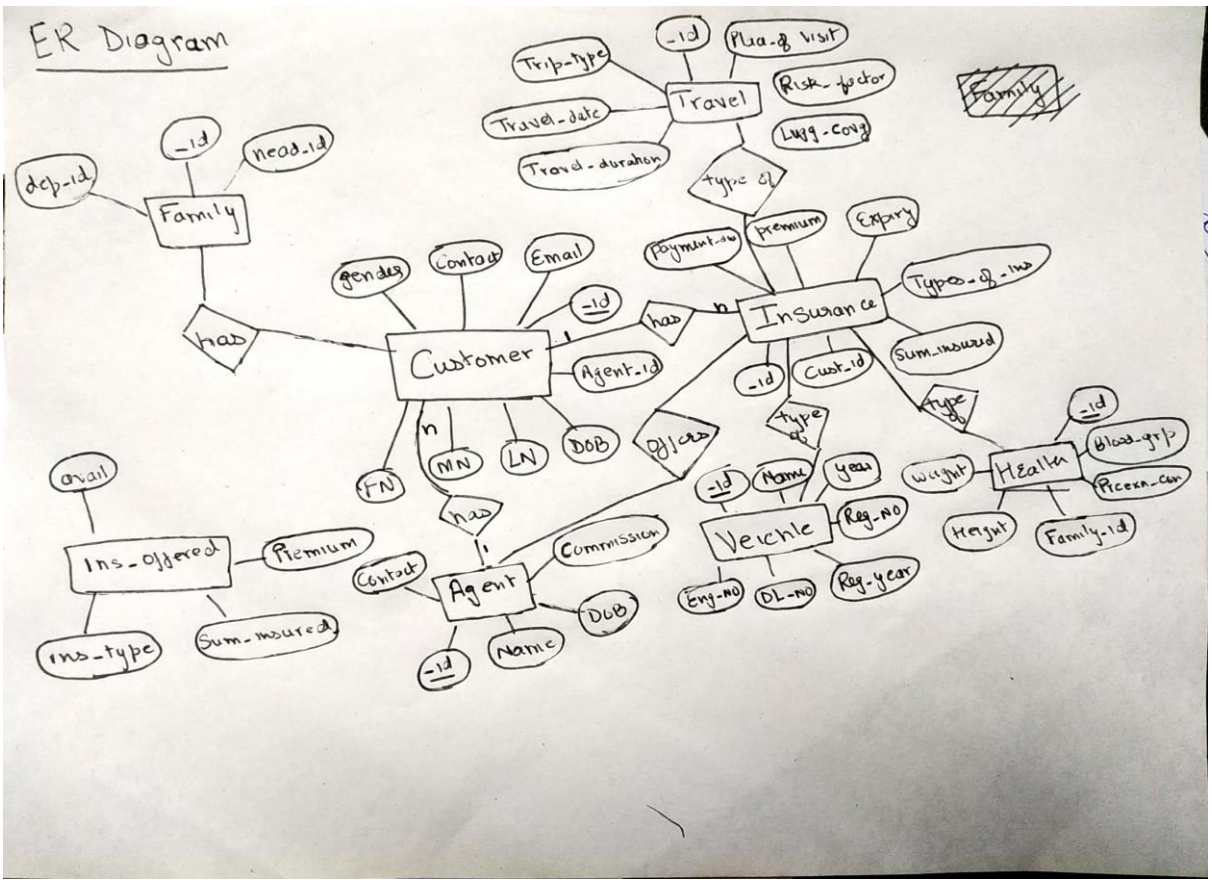
Literature Survey

In this section, literature survey is given. Accordingly, research papers are reviewed and analyzed based on the prediction methods used.

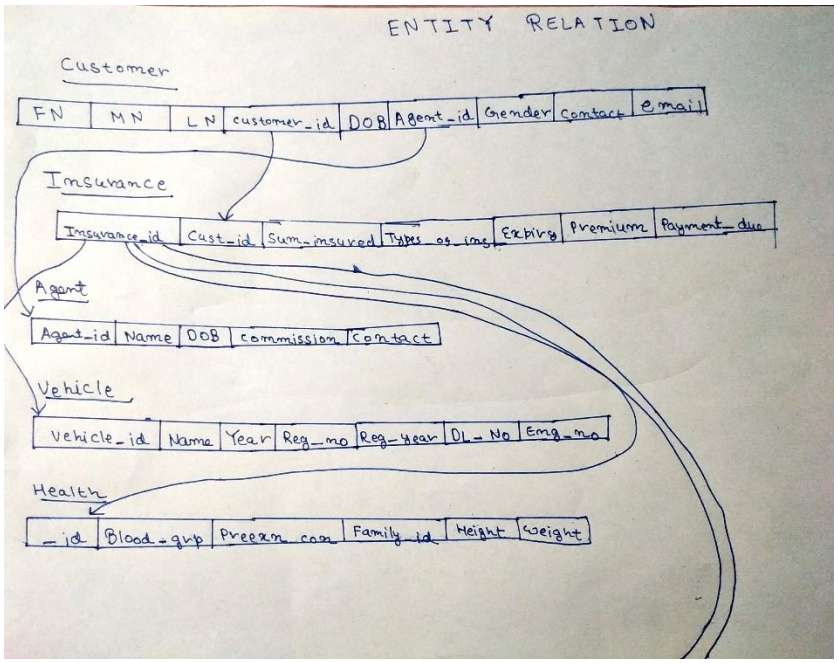
Authors	Method	Purpose	Advantages	Disadvantages
Momenul Ahmad	How to do web development	Giving general idea about web development and internet.	It provides essential knowledge to beginner.	Knowledge provided is limited and it is not helpful to a more experienced developer.
Joko Santoso	Building node applications with mongodb and backbone	Provides knowledge about mongo database	It is vast and contains wide range of examples.	It require a pre requisite knowledge and not recommended for beginner.
Kavya. S	1.1.1.1 A Study on Mongodb Database	Gives an overview on mongo database	Explains general functioning of mongo data base.	Doesn't contain practical examples.
Mazhar Ahmed	1.1.1.2 Kickstart with Node.js and MongoDB	Provides general idea about node js and mongo database	Contains a nice overview of node js and mongo database	It would have been better if it contain more examples and if was better organised.
Me Me Khaing, Kyi Pyar	1.1.1.3 Why Undergraduates Should Learn Web Development and Design Foundations	Provides basic knowledge of HTML and web	Provides HTML syntax and some examples regarding it.	It is nice document but it lacks some or the other syntax and contains less

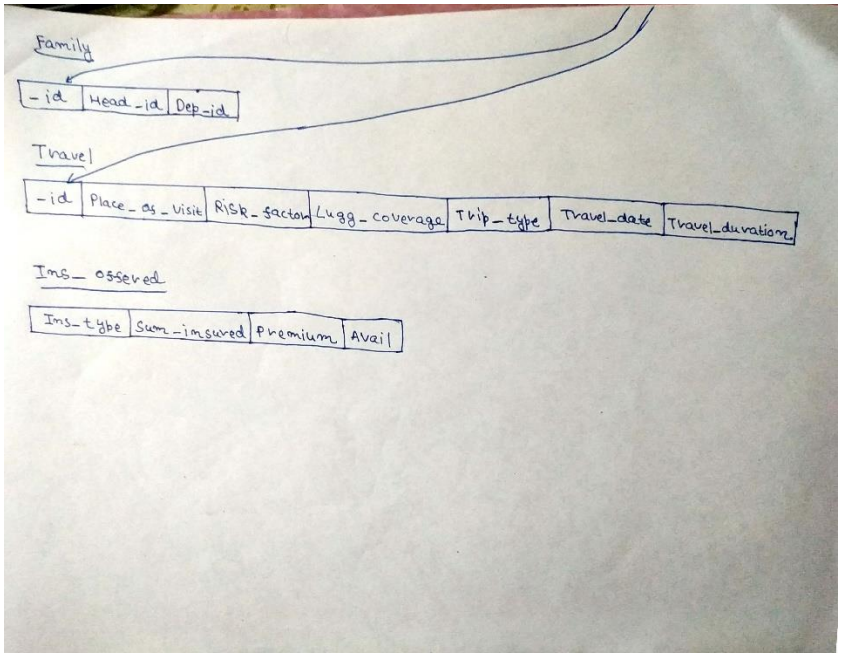
	with HTML5	development		examples.

ER Diagram:



Entity Relationship Model





List of Tables

Table: customer

Attribute	*Datatype*	*Constraint*
FN	Varchar2	Not null
MN	Varchar2	
LN	Varchar2	Not null
_id	Varchar2	Primary key
DOB	Date	
Agent_id	Varchar2	Foreign key to agent table
Gender	Varchar2	Sex in(M/F)
Contact	Number	Not null, Unique
Email	Varchar2	Not null, Unique

Table: Insurance

Attribute	*Datatype*	*Constraints*
_id	Varchar2	Primary key
Cust_id	Varchar2	Foreign key
Sum_insured	Number	Not null
Types_of_ins	Varchar2	Not null

Expiry	Date	Not null
Premium	Number	>0
Payment_due	Char	Y/N

Table: Agent

Attribute	*Datatype*	*Constraints*
_id	Varchar2	Primary key
Name	Varchar2	Not null
DOB	Date	Not null
Commission	Number	
Contact	Number	Not null, unique

Table: Vehicle

Attribute	*Datatype*	*Constraints*
_id	Varchar2	Primary Key
Name	Varchar2	Not null
Year	Varchar2	
Reg_no	Varchar2	Unique, Not null
Reg_year	Varchar2	
DL_No	Varchar2	Unique, Not null
Eng_no	Varchar2	Unique, Not null

Table: Health

Attribute	*Datatype*	*Constraints*
_id	Varchar2	Primary key
Blood_grp	varchar2	Not Null
Preexn_con	Varchar2	
Family_id	Varchar2	
Height	Number	Not Null, >0
Weight	Number	Not Null, >0

Table: Family

Attribute	*Datatype*	*Constraints*
_id	Varchar2	Primary key
Head_id	Varchar2	Not null
Dep_id	Varchar2	

Table: travel

Attribute	*Datatype*	*Constraints*
_id	Varchar2	Primary key
Place_of_visit	Varchar2	Not null
Risk_factor	Number	
Lugg_coverage	Char	Not null
Trip_type	Varchar2	

Travel_date	Date	
Travel_duration	Number	

Travel: Ins_offered

Attribute	*Datatype*	*Constraints*
Ins_type	Varchar2	
Sum_insured	Number	>0
Premium	Number	>0
Avail	Char	Y/N

Table: life_ins

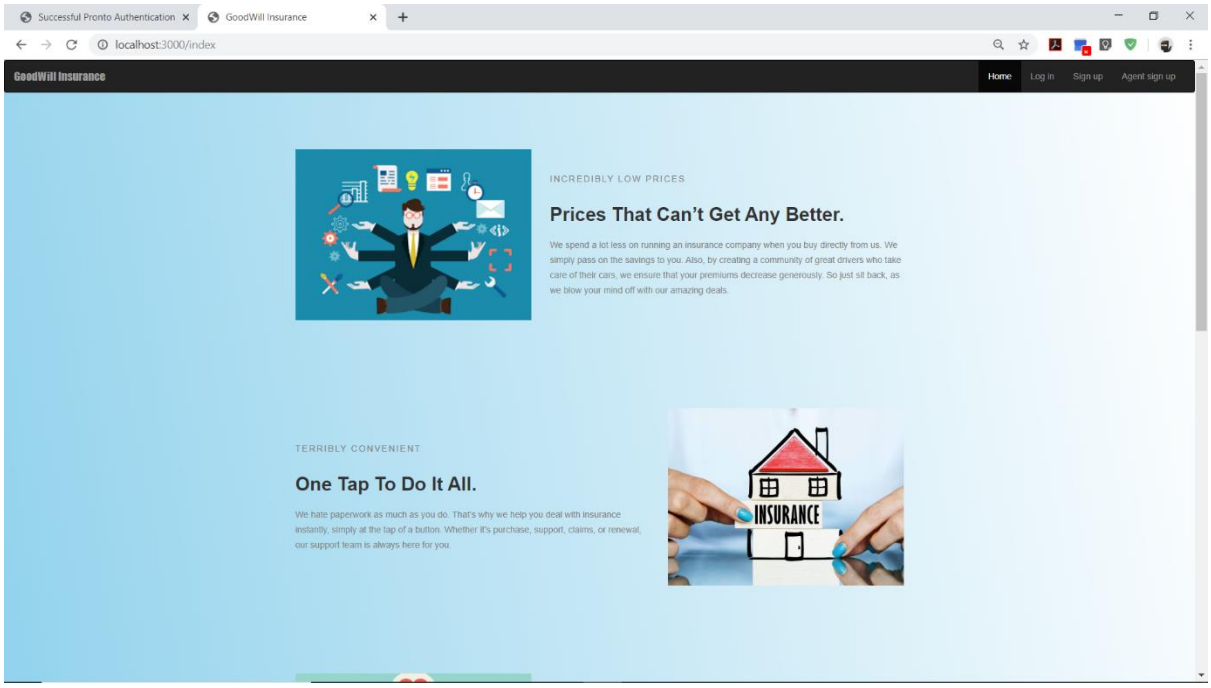
Attribute	*Datatype*	*Constraints*
_id	Varchar2	Primary key
Cover	Number	Not Null
Terms	Varchar2	
Nominee	Varchar2	Not Null
Nominee_rel	Varchar2	
Health_history	Varchar2	

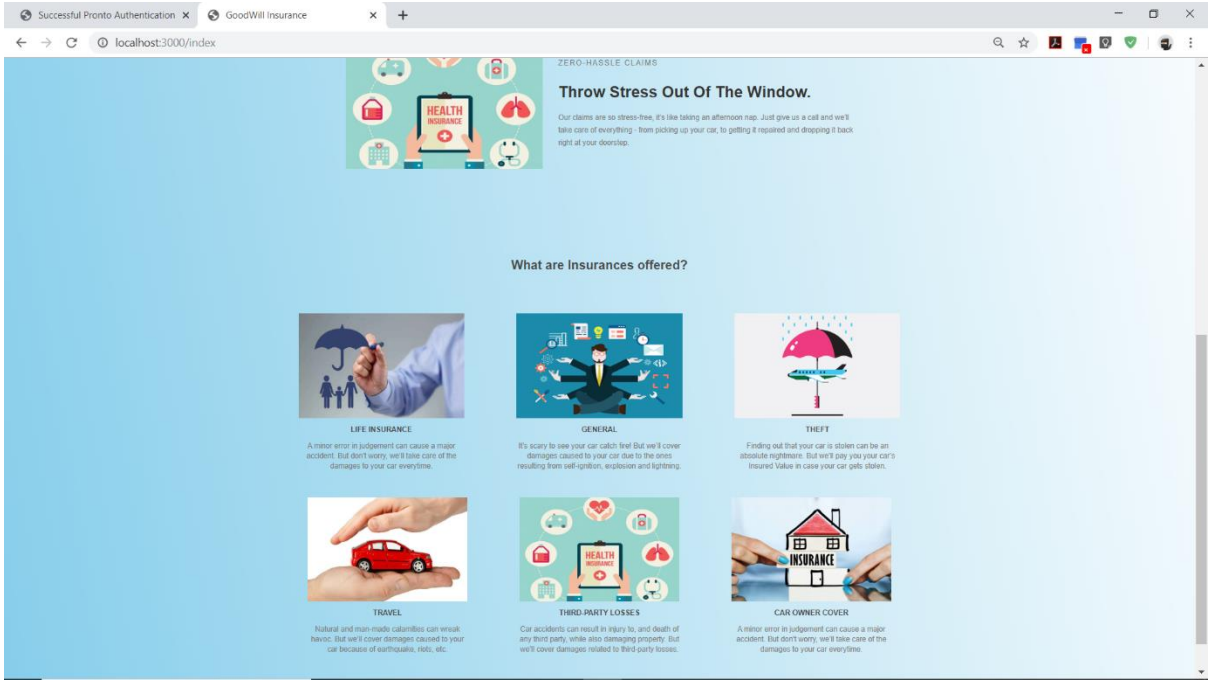
Work break down:

Team Member Registration Number	*Name*	*Work Assigned*
18BCI0190	Laksh Gupta	Frontend web dev, Mongo backend
18BCI0214	Mihir Srivastava	Frontend web dev, Mongo backend
18BCI0175	Ekansh Gupta	Mongo DB, Node JS connectivity
18BCI0188	Sumeet Roy Kurian	Node JS connectivity, Mongo DB

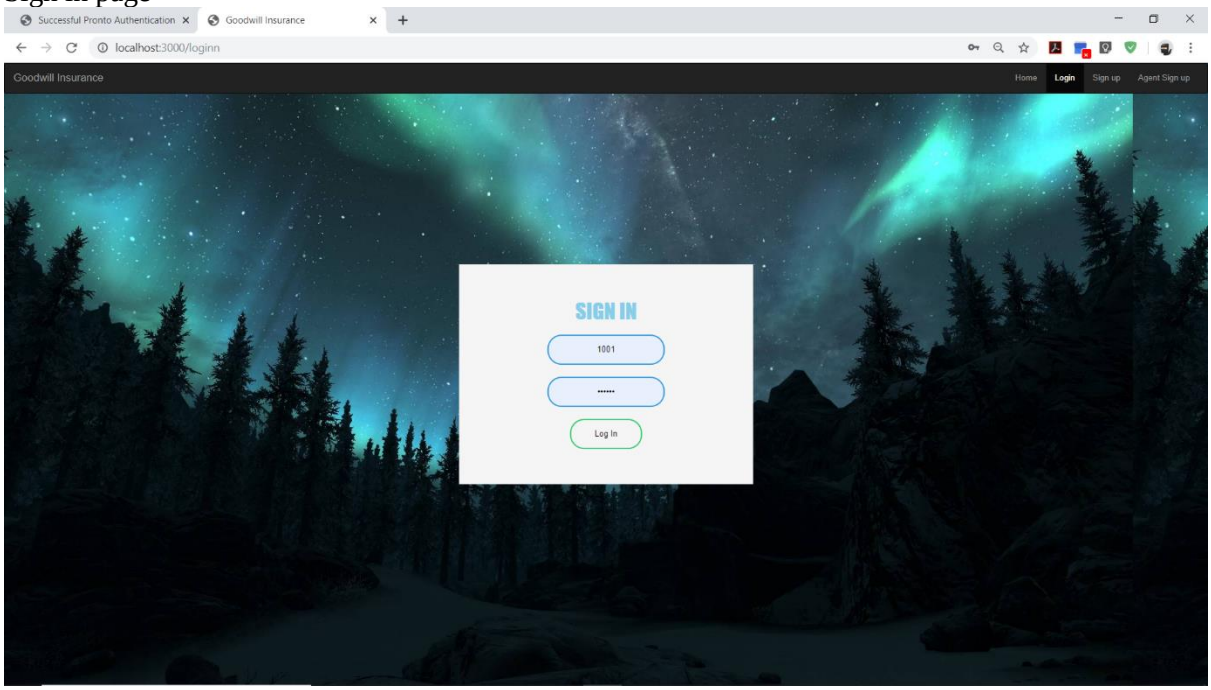
Screenshots

Home screen

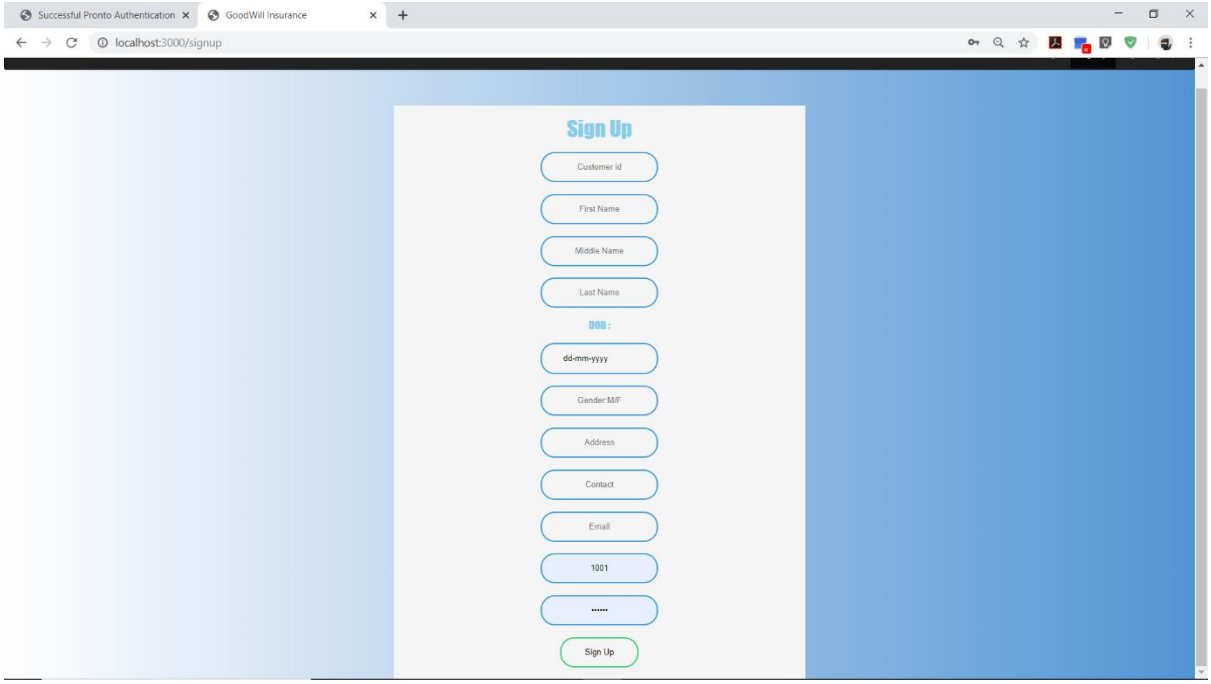




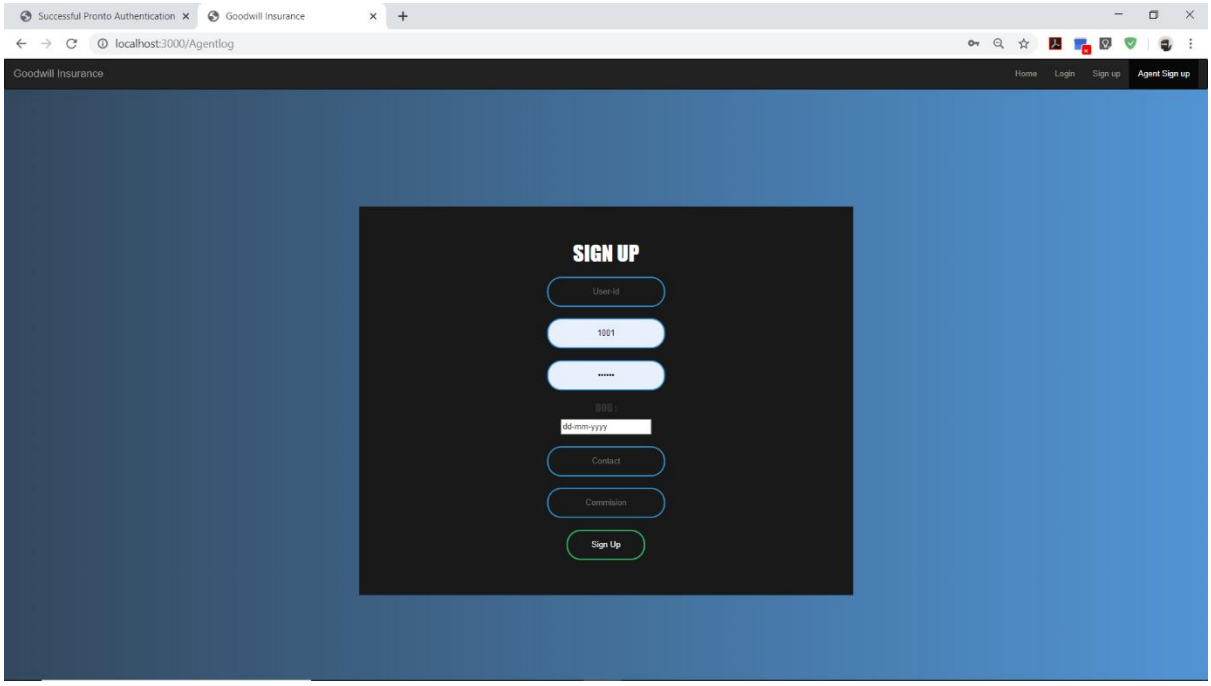
Sign in page



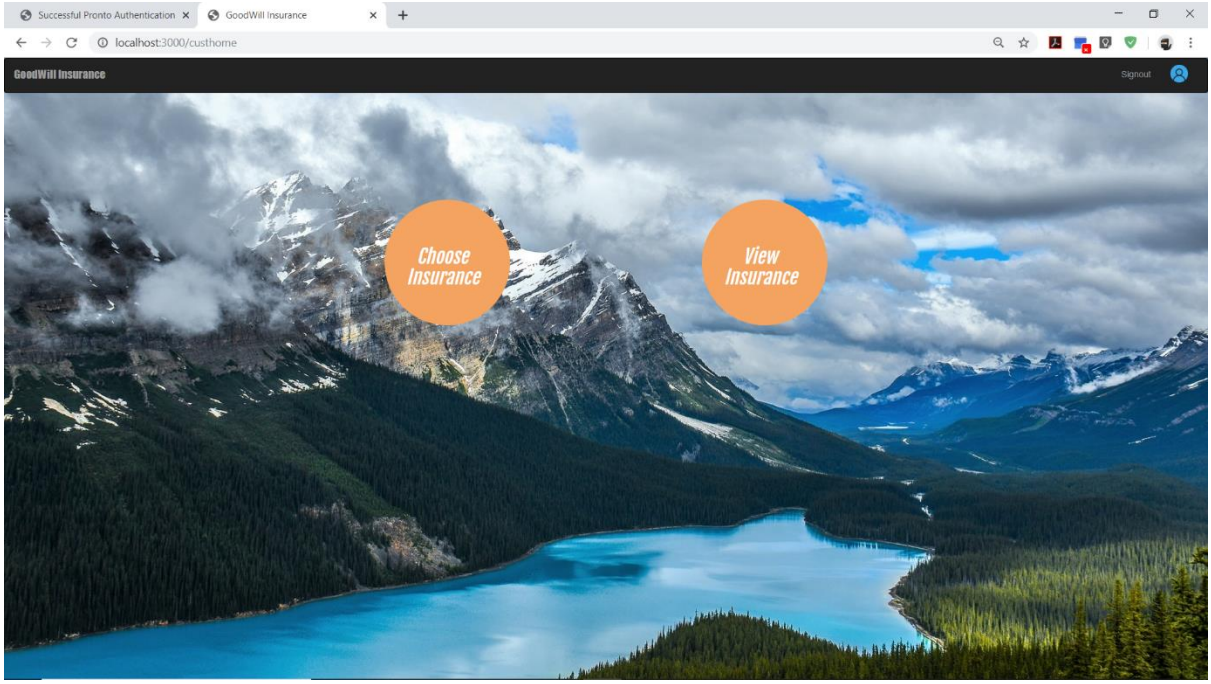
Sign up page-customer



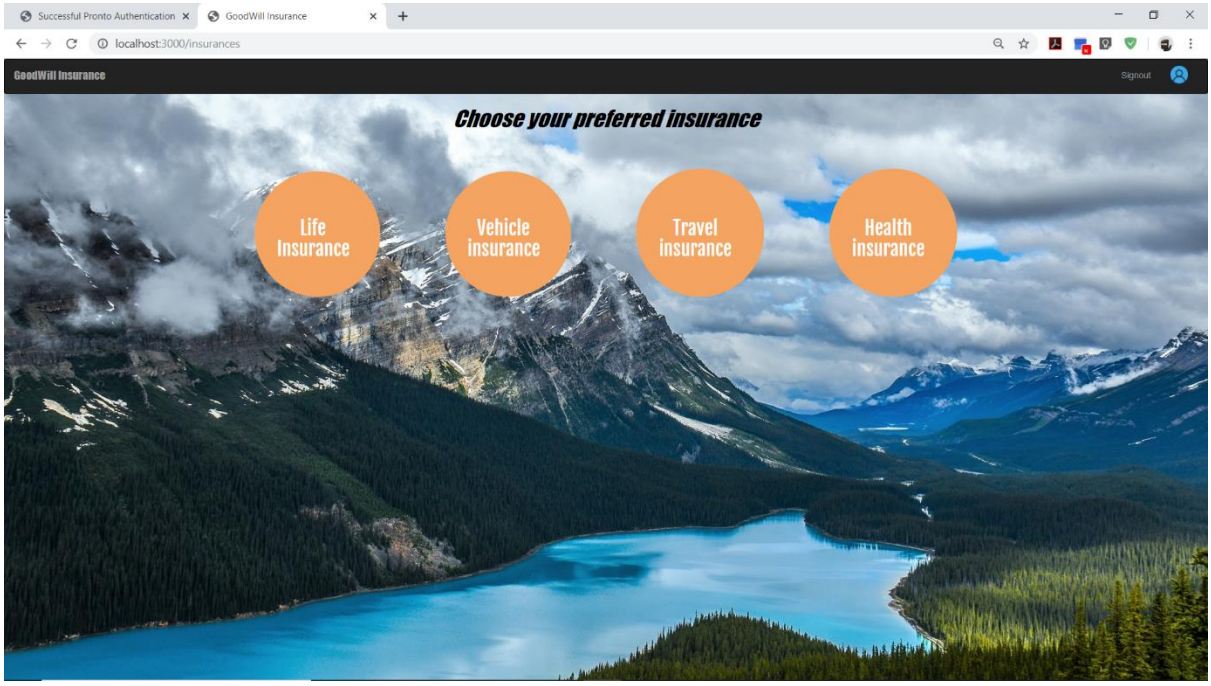
Agent sign up



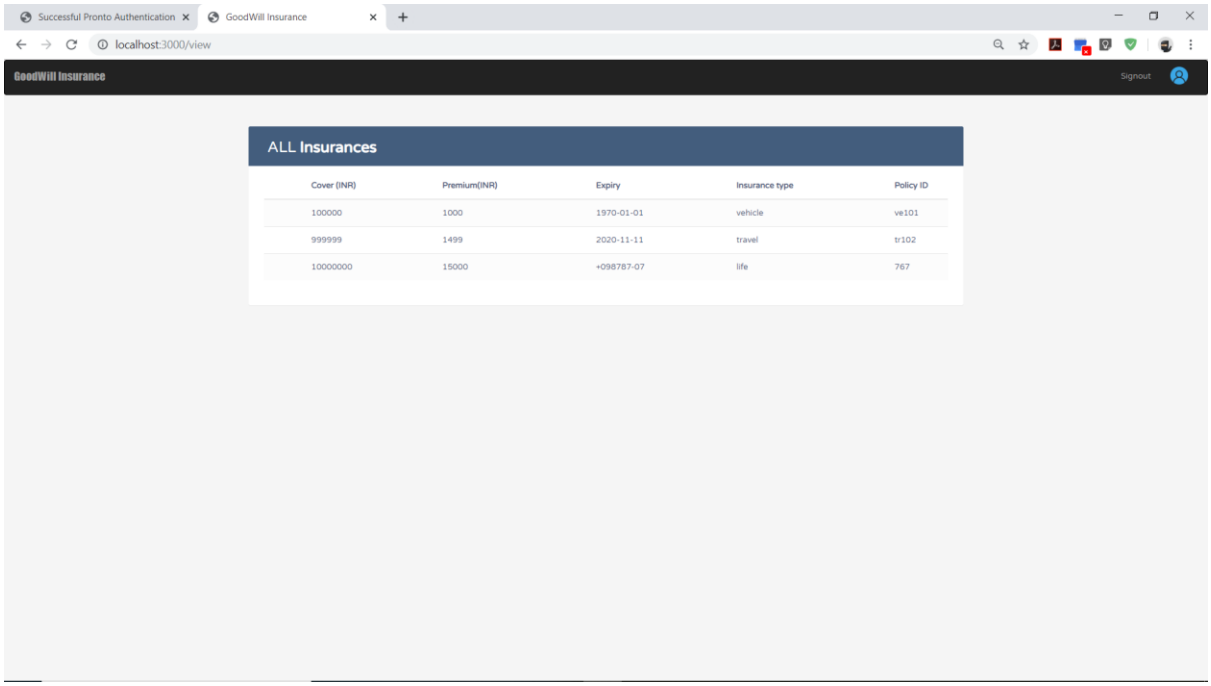
Home page for customer



Choosing new insurance for customer

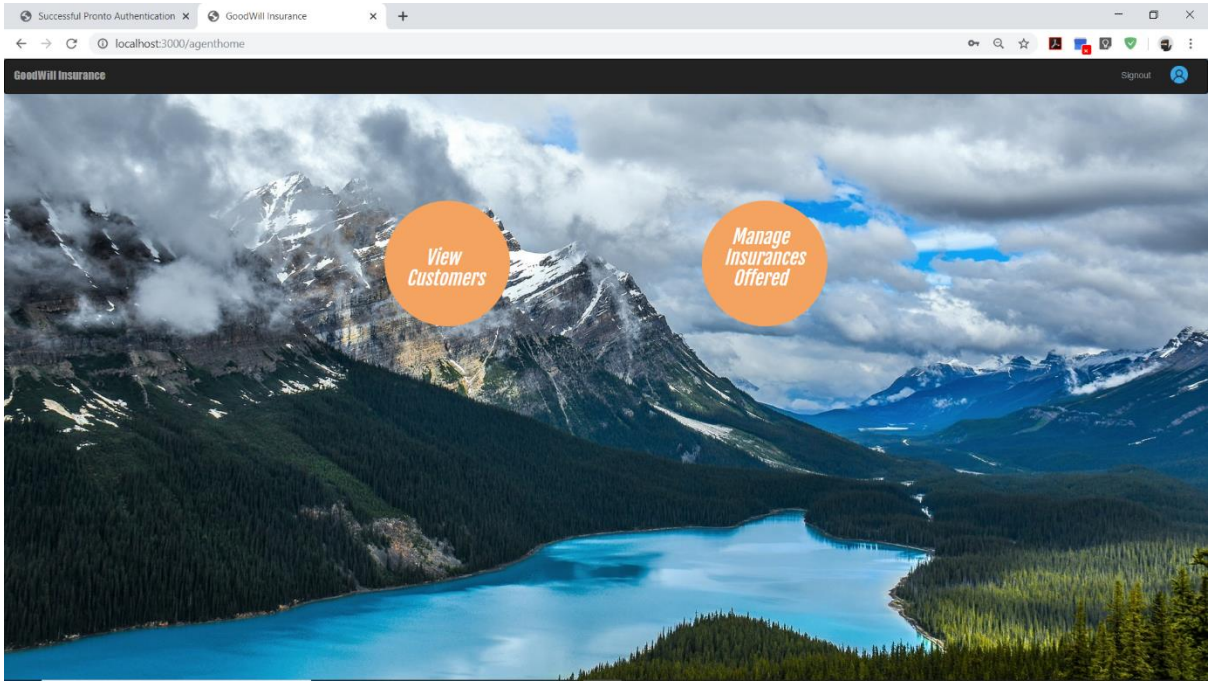


Viewing current insurances-customer

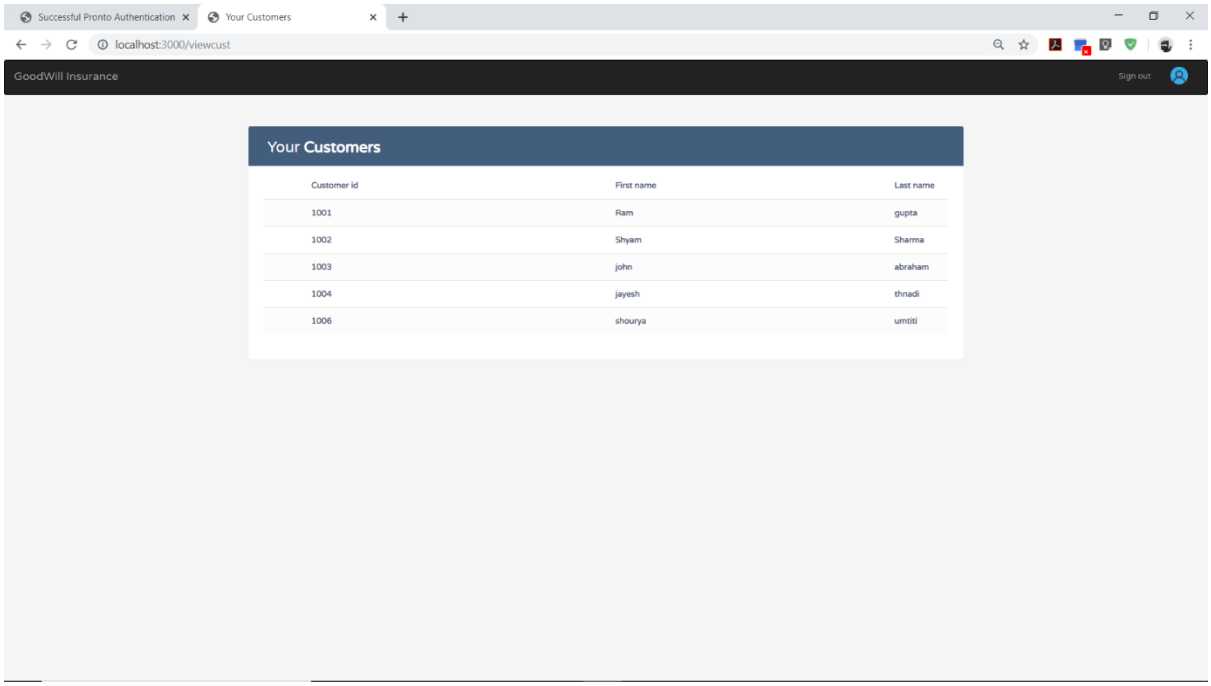


Cover (INR)	Premium(INR)	Expiry	Insurance type	Policy ID
100000	1000	1970-01-01	vehicle	ve101
999999	1499	2020-11-11	travel	tr102
10000000	15000	+098787-07	life	767

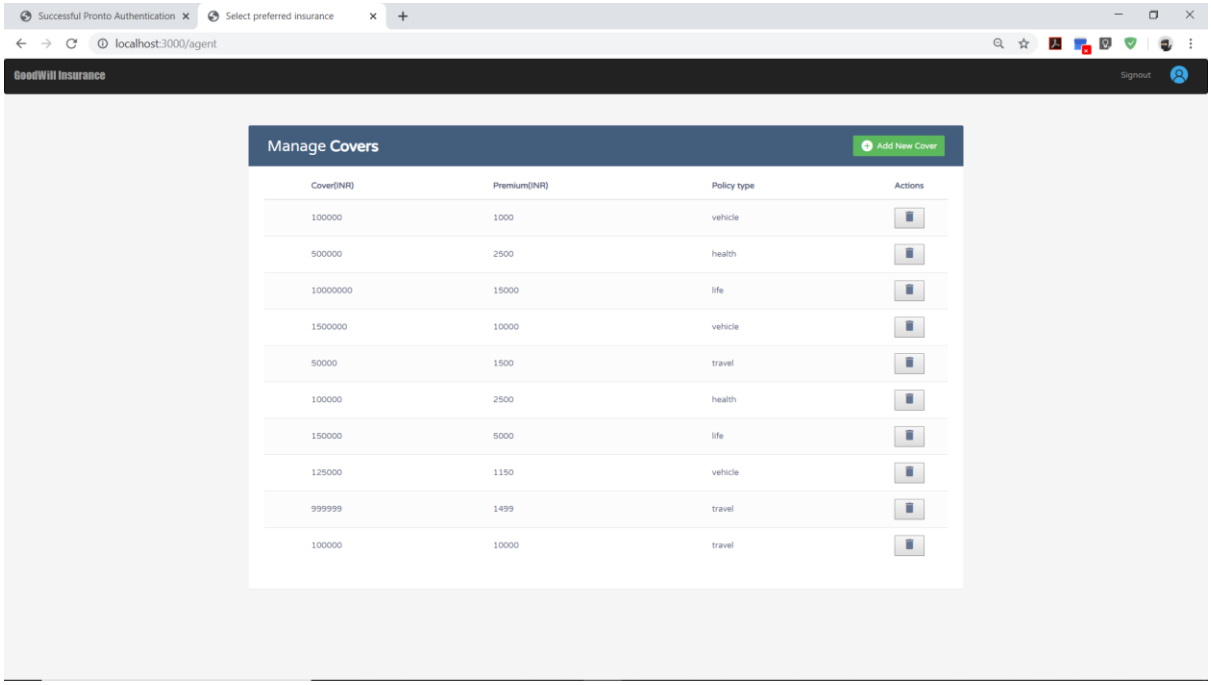
Agent home page- customer



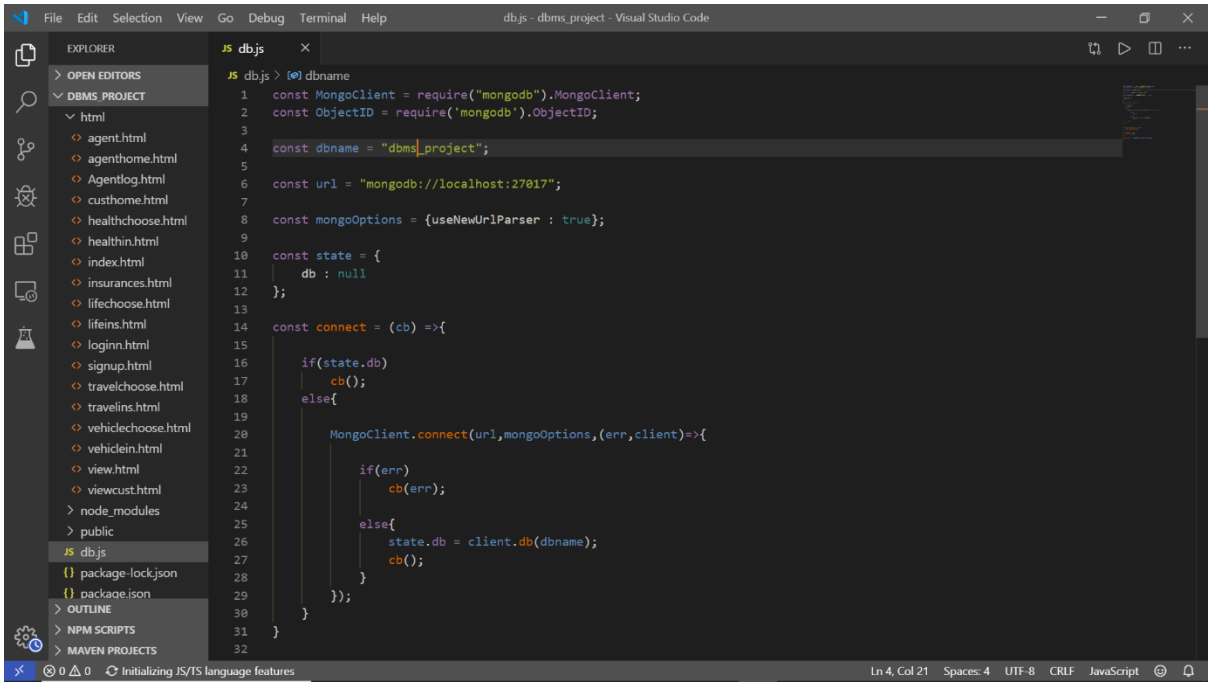
Viewing current insurances-agent



Manage insurances page-agent



Server connectivity JavaScript file



Server JavaScript file

```javascript
const express = require('express');
const bodyParser = require("body-parser");
const path = require('path');
const Joi = require('joi');

const db = require("./db");
const app = express();

const user = "user";
const customer = "customer";
const agent = "agent";
const ins = "ins_offered";
const insurance = "insurance";
const health = "health";
const travel = "travel";
const life = "life";
const vehicle = "vehicle";
const family = "family";

var userid="";
var cvr="";
var pre="";

app.use(bodyParser.json());
app.use(express.static('public'));
app.use(bodyParser.urlencoded({
    extended:true
}));

app.get('/',(req,res)=>{
    res.sendFile(path.join(__dirname,'html/index.html'));
});
```

Inserting data into database -JavaScript

```javascript
            res.json(documents);
        }
    });
});
app.get('/lifechoose_1',(req,res)=>{
    db.getDB().collection(ins).find({"ins_type" : "life"}).toArray((err,documents)=>{
        if(err)
            console.log(err);
        else{
            console.log(documents);
            res.json(documents);
        }
    });
});
app.get('/healthchoose_1',(req,res)=>{
    db.getDB().collection(ins).find({"ins_type" : "health"}).toArray((err,documents)=>{
        if(err)
            console.log(err);
        else{
            console.log(documents);
            res.json(documents);
        }
    });
});
app.get('/travelchoose_1',(req,res)=>{
    db.getDB().collection(ins).find({"ins_type" : "travel"}).toArray((err,documents)=>{
        if(err)
            console.log(err);
        else{
            console.log(documents);
            res.json(documents);
```

Posting data into database-JavaScript

```javascript
        console.log(documents);
        }
    });
});
app.post('/signup',function(req,res,next){
    var cust_id=req.body.cust_id;
    var fname=req.body.fname;
    var mname=req.body.mname;
    var lname=req.body.lname;
    var date=req.body.date;
    var gender=req.body.Gender;
    var address=req.body.Address;
    var contact=req.body.Contact;
    var email=req.body.email;
    var agent_id=req.body.agent_id;
    var pass=req.body.pass;
    userid = cust_id;
    var data1 = {
        "_id" : cust_id,
        "password" : pass
    }
    var data2 = {
        "_id" : cust_id,
        "F_name" : fname,
        "M_name" : mname,
        "L_name" : lname,
        "DOB" : new Date(date),
        "Gender" : gender,
        "Add" : address,
        "Cont" : contact,
        "Email" : email,
        "agent_id" : agent_id
```

Connection to database -JavaScript

```javascript
    }
});

app.delete('/ins/:id',(req,res)=>{
    console.log("in delete");
    const insID = req.params.id;
    console.log(insID);
    db.getDB().collection(ins).findOneAndDelete({_id : db.getPrimaryKey(insID)},(err,result)=>{
        if(err)
            console.log(err);
        else{
            res.json(result);
            console.log(result);
        }
    });
});

db.connect((err)=>{
    if(err){
        console.log('unable to connect to database');
        process.exit(1);
    }

    else{
        app.listen(3000,()=>{
            console.log('connected to database, app listening on port 3000');
        });
    }
});
```

Getting input-HTML