A Project report on

ONLINE IDE

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the academic requirements for the award of the degree.

Bachelor of Technology

in

Computer Science and Engineering

Submitted by

P. Shiva Sai (19H51A05L5)
G. Vikas Reddy (19H51A05N3)
R. Abhinav (19H51A05L6)

Under the esteemed guidance of
Mr. M. Shiva Kumar
Assistant Professor

Department of Computer Science and Engineering

CMR COLLEGE OF ENGINEERING & TECHNOLOGY
(An Autonomous Institution, Approved by AICTE, Affiliated to JNTUH, NAAC 'A+')
KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

2019- 2023

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project report entitled **"ONLINE IDE"** being submitted by **P. Shiva Sai** ***(19H51A05L5),*** **G. Vikas Reddy** ***(19H51A05N3),*** **R. Abhinav** ***(19H51A05L6)*** in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out his/her under my guidance and supervision.

The results embodies in this project report have not been submitted to any other University or Institute for the award of any Degree.

Mr. M. Shiva Kumar
Assistant Professor
Dept. of CSE

Dr. S. Siva Skanda
Associate Professor and HOD
Dept. of CSE

Acknowledgement

With great pleasure I want to take this opportunity to express my heartfelt gratitude to all the people who helped in making this project work a grand success.

I am grateful to **Mr. M. Shiva Kumar,** Assistant Professor , Dept of Computer Science and Engineering for his valuable suggestions and guidance during the execution of this project work.

I would like to thank **Dr. S. Siva Skanda**, Head of the Department of Computer Science and Engineering, for his moral support throughout the period of my study in CMRCET.

I am highly indebted to **Dr. V A Narayana**, Principal CMRCET for giving permission to carry out this project in a successful and fruitful way.

I would like to thank the Teaching & Non- teaching staff of Department of Computer Science and Engineering for their co-operation

Finally I express my sincere thanks to **Mr. Ch. Gopal Reddy**, Secretary, CMR Group of Institutions, for his continuous care. I sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project work.

P. Shiva Sai - 19H51A05L5
G. Vikas Reddy - 19H51A05N3
R. Abhinav - 19H51A05L6

TABLE OF CONTENTS

LIST OF FIGURES

Abstract

The name of our project is Online IDE. The main purpose of this project is to build a online compiler which can compile 5 languages C, C++, java, python and java script. We have many online compiler, but we cannot store our codes in it. In this, we can create an account and store all our codes. We can generate link for our program and share it with others. It is done using java script, mongo dB, react js, node js, pistonapi, material Ui and google auth technologies.

CHAPTER 1

INTRODUCTION

1.1 DESCRIPTION :-

- A compiler is a special program that translates a programming language's source code into machine code, bytecode or another programming language.
- We have many compilers available in which some of them are online and some are offline.
- Offline compilers are permitted to only one language for which we need to have a text editor, terminal as well as tools to compile the language.
- Online compilers are better compared to offline compilers but we cannot save our code or share our code in these. So it would be more easier for competitive coders if they have a online compiler in which they can save the code as well as share the code.

1.2 SCOPE :-

- This application helps user to code and compile in 5 different languages and save them in their profile.
- Users need not share the codes in the form of files, they can share in the form link.

1.3 OBJECTIVE :-

- The main objective of our project is to create a personal online IDE for competitive coder.
- In fact, user should feel comfortable with the user interface and utilize the benefits of IDE, it need to be used as compiler as well as editor.
- We want to use a simple architecture and use less tools to build so that it makes compiler more secured compared to others

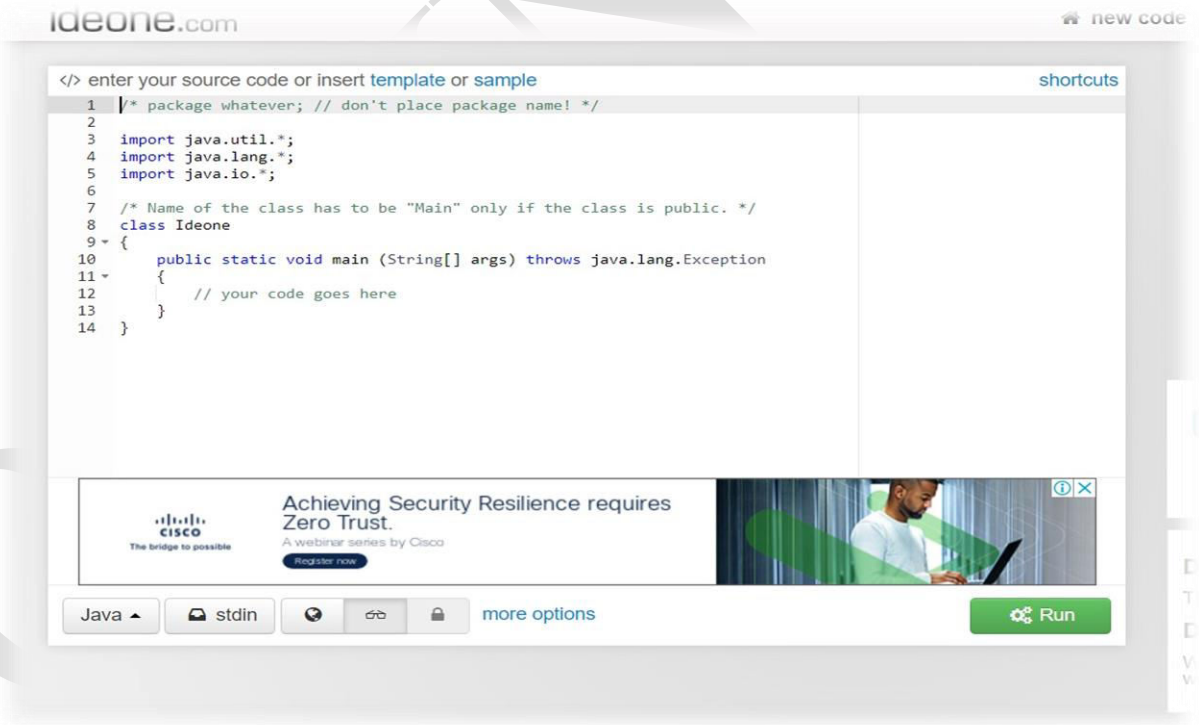
CHAPTER 2

BACKGROUND WORK

2.1 EXISTING SOLUTION

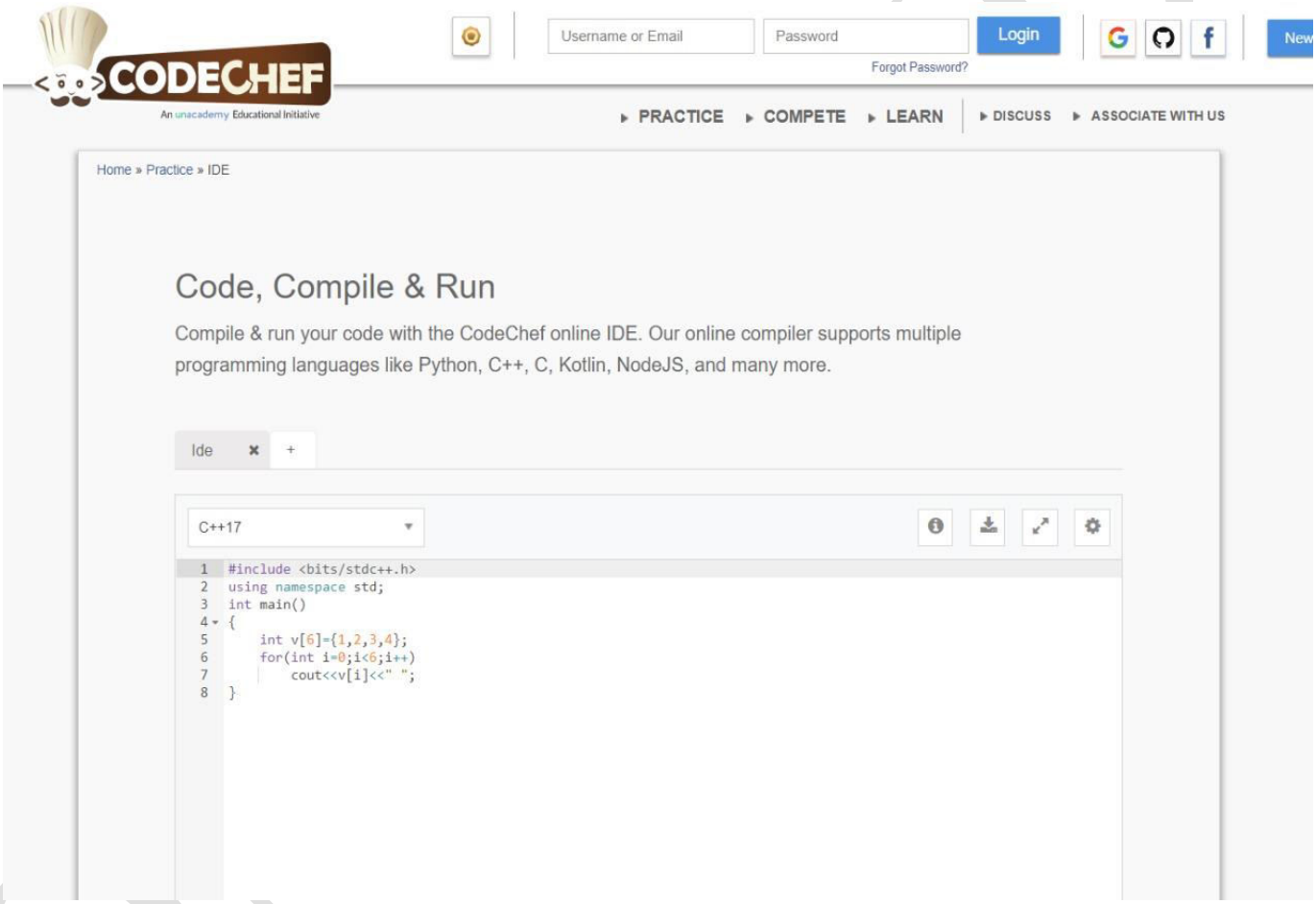
Ideone :-

- Ideone is an online compiler and debugging tool which allows user to compile source code and execute it online in many programming languages.
- We can also share our code in ideone.
- But we cannot save code and whenever page gets refreshed we need to write our code again.



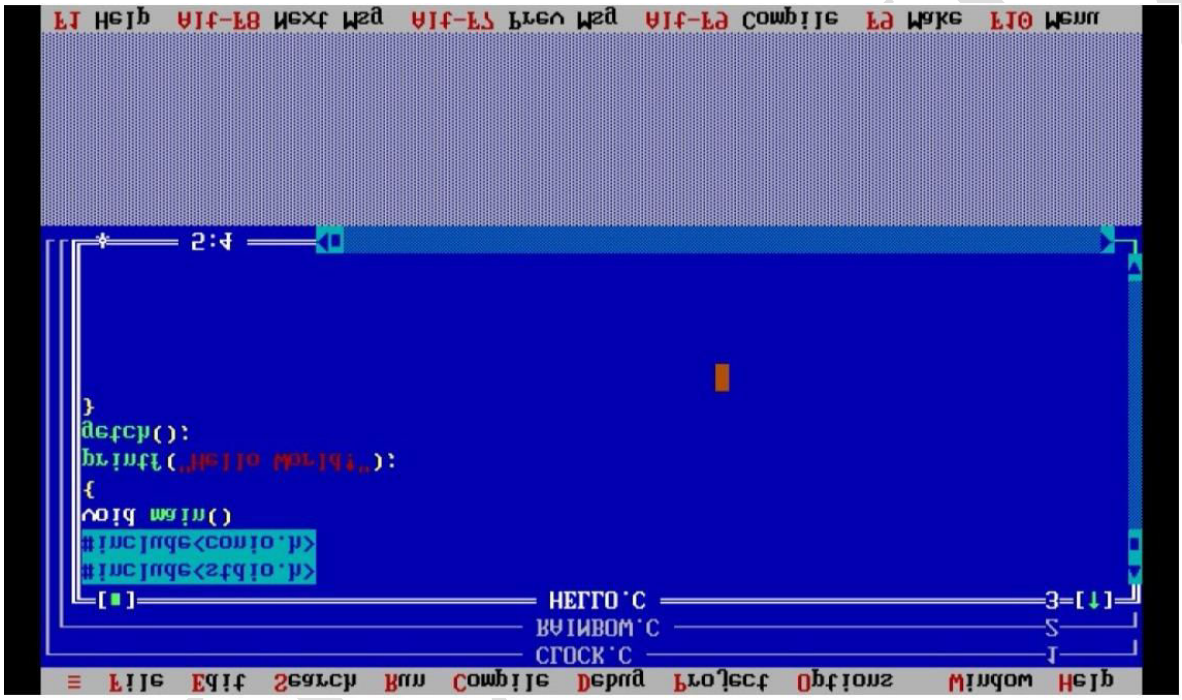
Codechef :-

- Code-Chef is an online educational program and competitive programming community of global programmers.
- Every week there will be different competitions om code-chef and it gives rating to its coders.



Turbo C :-

- Turbo C is a discontinued integrated development environment and compiler for the C programming language.
- This can be only used for c programming.
- It saves the code in files so to share the code we need to send complete file.



2.2 DISADVANTAGES :-

- Most of the offline compilers are permitted to only one language.
- Most of the online compilers have complicated architecture.
- In most of the online compilers we can neither save the code nor share the code in less complexity.

CHAPTER 3

PROPOSED SYSTEM

3.1) PROPOSED SOLUTIONS :-

Our idea is to create a online IDE. Users can login in it using their google accounts. Here user is given a separate id. User need to select the language and type the code in the editor. After writing the code he can save the code or run the code. The code is saved in his account and he can view it whenever he login. If he run the code all these input is sent to piston api and piston api compiles the code and sends the output. The output sent by the api is displayed in the output window. Here we have given another option to user to generate link of their code. This generates a link by which he can share his code. The technologies used are mongo dB, react js, node js, piston api, material Ui and google auth api.

3.1.1) Advantages of proposed solutions :-

- Code can be shared in user profile.
- Simple architecture.
- Code sharing is done by link.

3.2) REQUIREMENTS :-

Our project includes two types of system requirements:

- Hardware
- Software

3.2.1) Requirements for hardware :-

- 64-bit Microsoft® Windows® 8/10.
- x86_64 CPU architecture. 2nd generation Intel Core or newer.
- 4 GB RAM or more.
- 1280 x 800 minimum screen resolution.

3.2.2) Requirements for Software :-

- Node js
- VS Code
- Piston API
- Mongodb

3.3) DESCRIPTION :-

Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on a JavaScript Engine and executes JavaScript code outside a web browser, which was designed to build scalable network applications. MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server Side Public License which is deemed non-free by several distributions.Visual Studio Code, also commonly referred to as VS Code, is a source-code editor made by Microsoft with the Electron Framework, for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.Piston is a high performance general purpose code execution engine. It excels at running untrusted and possibly malicious code without fear from any harmful effects.

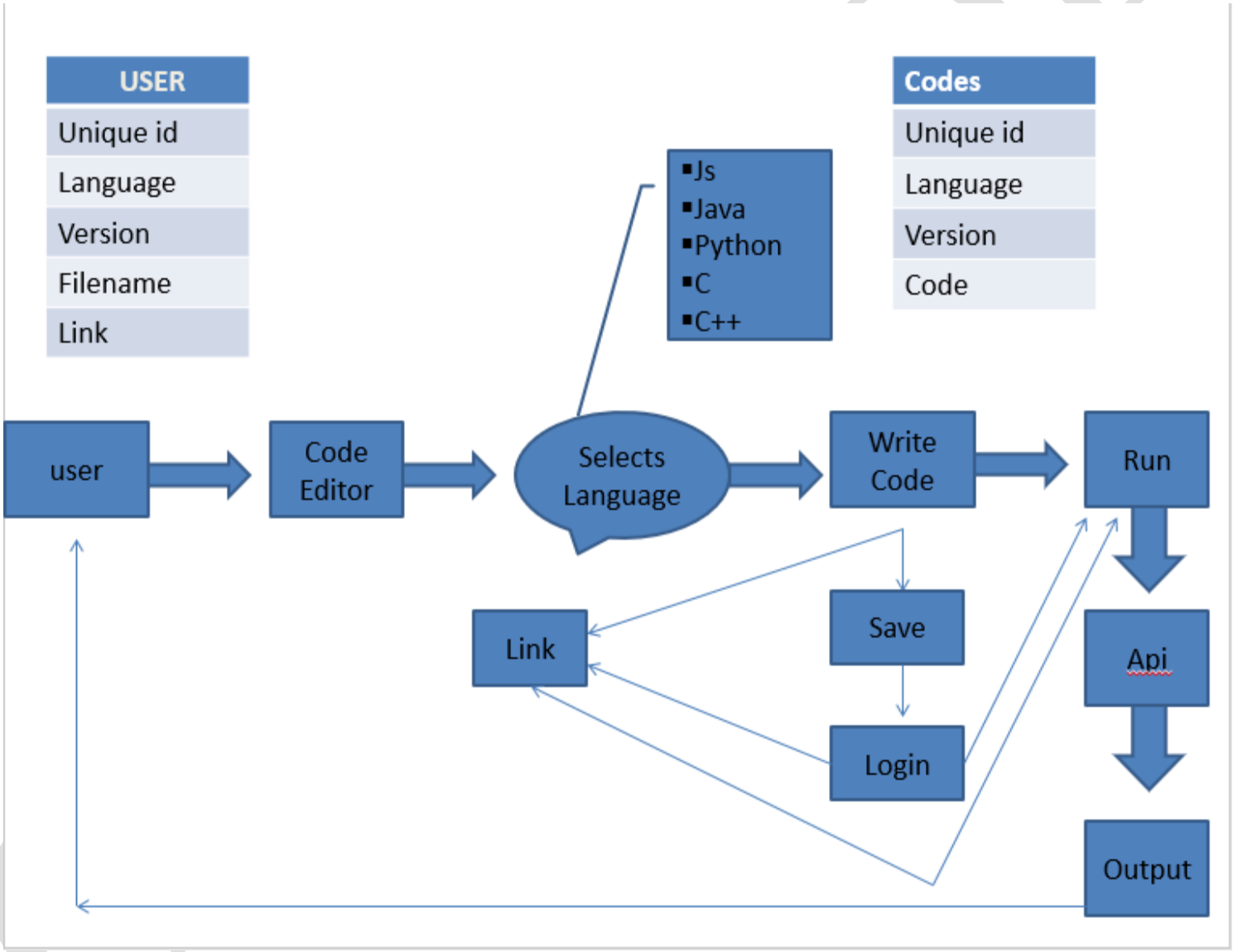
3.4) APP CONFIGURATION :-

- Node version :- v16.9.0
- VS Code version :- 1.72
- Mongodb version :- 6.0
- Size :- 16 MB

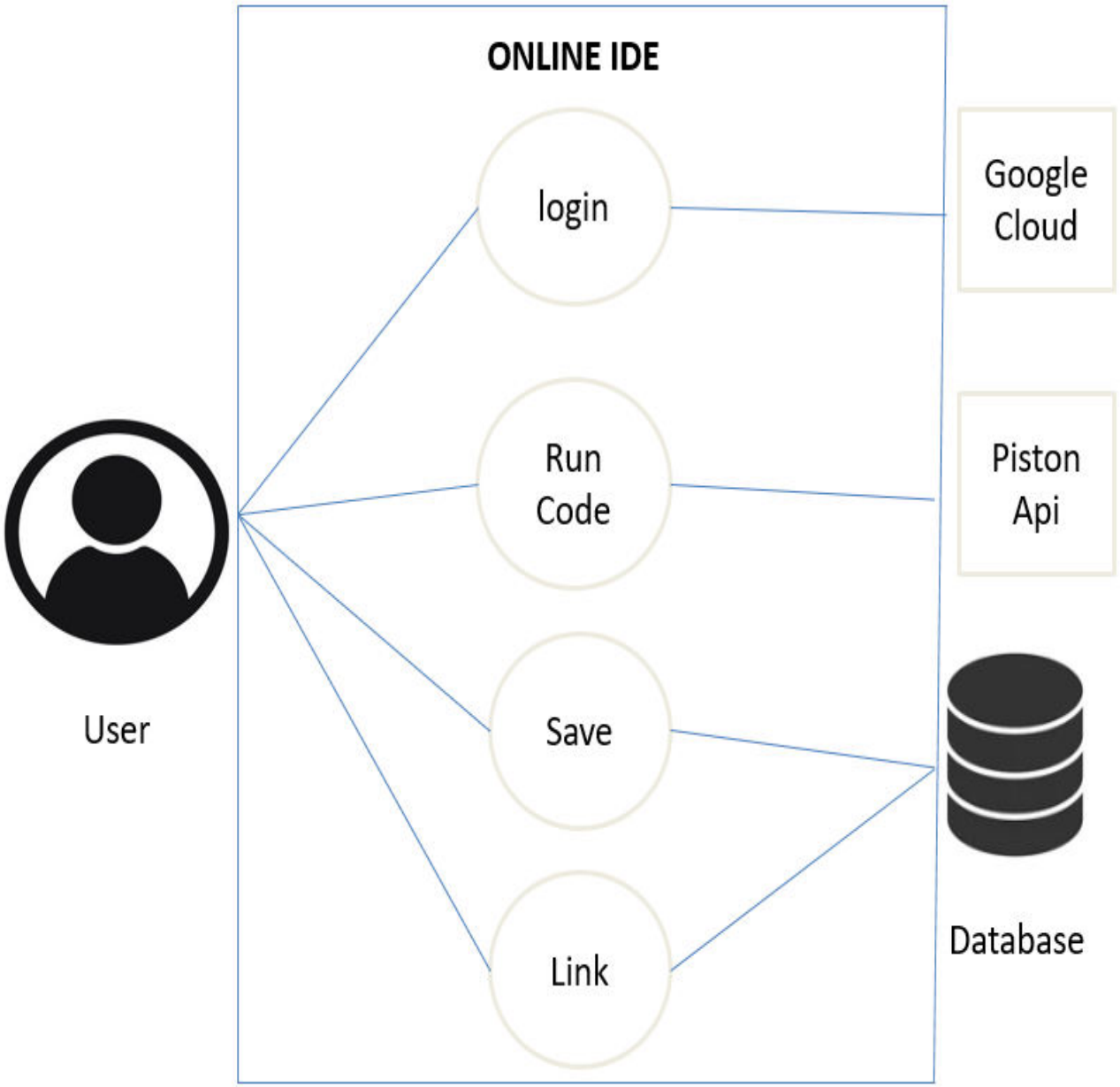
CHAPTER 4

DESIGNING

4.1 BLOCK DIAGRAM :-



4.2 ARCHITECTURE DIAGRAM :-



CHAPTER 5

RESULTS AND DISCUSSION

5.1) RESULTS AND DISCUSSIONS :-

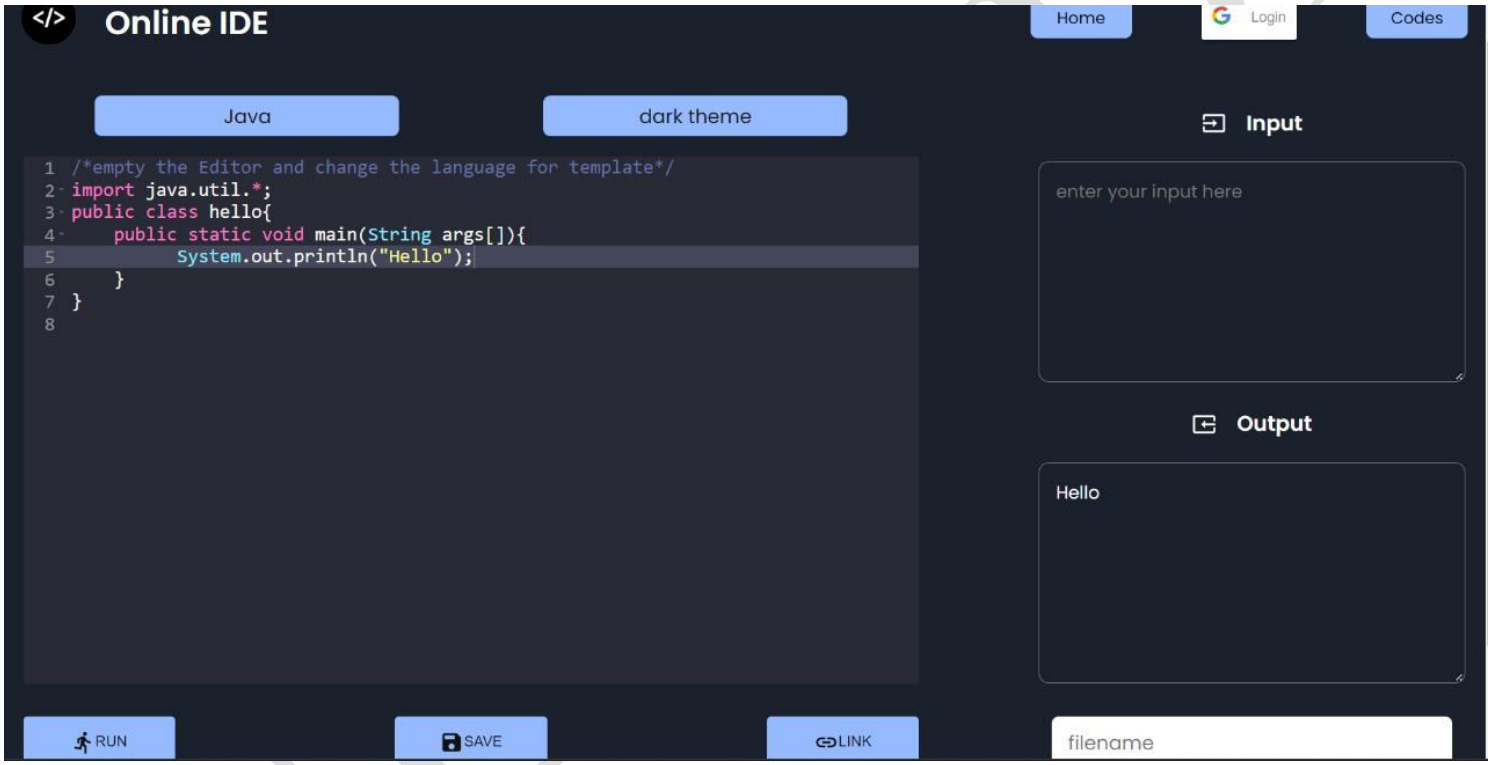


Fig: Output

Fig: User Input

Fig: Saved Codes

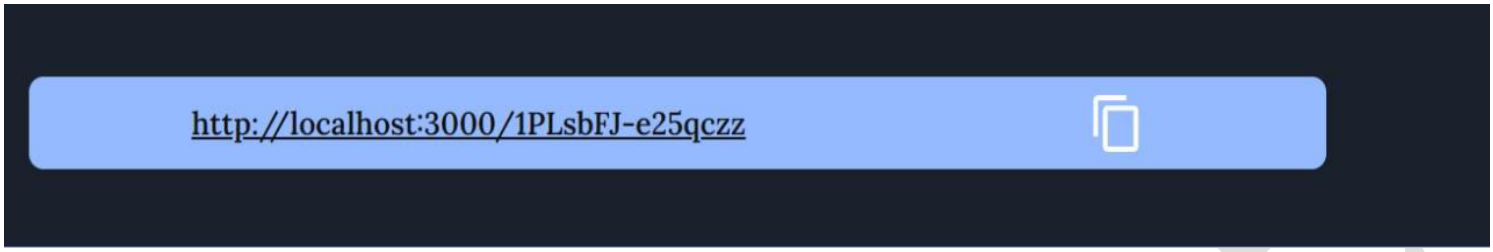


Fig: Generated Link

5.2) Source code :-

i) Front end :-

iblocks-frontend > src > JS App.js > ...

```
1   import './App.css';
2   import React from 'react';
3   import Editor from './components/Editor';
4   import {BrowserRouter as Router,Route} from "react-router-dom";
5   import Nav from './components/Nav';
6   import Codes from './components/Codes';
7
8
9   function App() {
10    console.log("appjs")
11    return (
12      <div className="app">
13      <Router>
14          <Nav/>
15          <Route path="/" exact component={Editor}/>
16          <Route path= "/stored/codes" exact component={Codes}/>
17          <Route path="/:id" exact component={Editor}/>
18      </Router>
19      </div>
20    );
21  }
22
23  export default React.memo(App);
```

Fig: App.js

iblocks-frontend > src > JS index.js > ...

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';

ReactDOM.render(
  <App/>,
  document.getElementById('root')
);

const  button = document.getElementById('burger');
const nav = document.getElementById('nav_right');
button.addEventListener('click',()=>{
  console.log('clicked')
  nav.classList.toggle('nav_rightexpanded');
});

const link = document.querySelectorAll('.navlink');
link[0].addEventListener('click',()=>{
  nav.classList.toggle('nav_rightexpanded');
});

link[1].addEventListener('click',()=>{
  nav.classList.toggle('nav_rightexpanded');
});
```

Fig: Index.js

iblocks-frontend > src > JS templates.js > ...

```
const java = `/*empty the Editor and change the language for template*/
import java.util.*;
public class hello{
    public static void main(String args[]){

    }
}
`
const cpp_competitive = `/*empty the Editor and change the language for template*/
#include<bits/stdc++.h>
#define ll long long
using namespace std;
void solve()
{

}
int main()
{
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    int t;
    cin>>t;
    while(t--)
    {
        solve();
    }
    return 0;
}
`
```

Fig: Template.js

ii) Backend :-

iblocks-backend > JS model.js > ...

```
1   import mongoose from "mongoose";
2
3   const user = new mongoose.Schema({
4       uniqueid:String,
5       language:String,
6       version:String,
7       filename:String,
8       link:String
9   });
10
11  const codes = new mongoose.Schema({
12      uid:String,
13      language:String,
14      version:String,
15      code:String,
16  });
17
18  const User = mongoose.model('user',user);
19  const codefile = mongoose.model('code',codes);
20
21  export default User;
22  export {codefile};
```

Fig: Model.js

ONLINE IDE

iblocks-backend > JS server.js > ...

```
/*npm dependencies */
import dotenv from 'dotenv';
import express from "express";
import cors from "cors";
import bodyParser from "body-parser";
import mongoose from "mongoose";
import user,{codefile} from "./model.js";
import { nanoid } from "nanoid";
dotenv.config();
/*app config*/
const app = express();
app.use(bodyParser.json({extended:1}));
app.use(cors({
    origin:'http://localhost:3000'
}));
app.use(bodyParser.urlencoded({extended:1}));

var gid="";
/*mongoose setup */
const mogoose_connection_url = "mongodb+srv://shivasai:shivasai@cluster0.nrpiasi.mongodb.net/?retryWrites=true&w=majority"
mongoose.connect(mogoose_connection_url,{useNewUrlParser:true,useUnifiedTopology: true});

/*constants in app */
const port = process.env.PORT||5000;


app.get("/",(req,res)=>{
    res.send("<h1>running succesfully</h1>");
})
app.post("/fetchcodes",(req,res)=>{

    const uid = req.body.uniqueid;
    gid = uid;
    user.find({uniqueid:uid},(err,result)=>{
        if(err)
```

iblocks-backend > JS server.js > ...

```
        if(err)
        {
            res.status(400).json({message:"internal database error"});
        }
        else
        {
            if(result)
            {
                res.status(200).send(result);
            }
            else
            {
                res.status(200).json({data:[]});
            }
        }
    })
});

app.get("/unqid",(req,res)=>{
    const id = nanoid(15);
    if(id===undefined)
    {
        res.status(400).json({message:"unabe to generate link"})
    }
    else{
        res.status(200).json({link:id});
    }
});

app.post("/:id",(req,res)=>{
    const id = req.params.id;
    if(id==="save")
    {
        user.findOne({uniqueid:req.body.uniqueid,link:req.body.link},(err,result_user)=>{
            if(err){
                res.status(400).json({message:"internal databse error"});
            }
```

iblocks-backend > JS server.js > ...

```
72                  else
73                  {
74                      if(result_user)
75                      {
76                          res.status(200).json({saved:false});
77                      }
78                      else{
79                       const save_data = new user({
80                          uniqueid:req.body.uniqueid,
81                          language:req.body.language,
82                          version:req.body.version,
83                          filename:req.body.filename,
84                          link:req.body.link
85                      });
86                      save_data.save();
87                      res.status(200).json({saved:true});
88                  }
89                  }
90              })
91
92          }
93          else{
94          codefile.findOne({uid:id},(err,result)=>{
95              if(err)
96              {
97                  res.status(400).json({message:"internal database error"});
98              }
99              else
100             {
101                 if(result){
102                     res.status(200).json({found:true});
103                 }
104                 else
105                 {
106                     const code_file = new codefile({
107                         uid:id,
108                         language:req.body.language,
104                  else
105                  {
106                      const code_file = new codefile({
107                          uid:id,
108                          language:req.body.language,
109                          version:req.body.version,
110                          code:req.body.code
111                      });
112                      code_file.save();
113                      res.status(200).json({message:true});
114                  }
115              }
116          });
117          }
118      })
119      app.get("/:id",(req,res)=>{
120          const unqid = req.params.id;
121          codefile.findOne({uid:unqid},(err,result)=>{
122              if(err)
123              {
124                  res.status(400).json({message:"internal databse error"});
125              }
126              else
127              {
128                  if(result)
129                  {
130                      res.status(200).json(result);
131                  }
132                  else{
133                      res.status(200).json({message:false});
134                  }
135              }
136          })
137      });
```

```
app.post("/codes/language",(req,res)=>{
    const lan = req.body.language;
    user.find({uniqueid:gid,language:lan},(err,results)=>{
        if(err)
        {
            res.status(400).json({message:false});
        }
        else
        {
            res.status(200).send(results);
        }
    });
});
app.post("/code/delete",(req,res)=>{
    const id = req.body.id_to_delete;
    user.deleteOne({_id:id},(err,result)=>{
        if(err)
        {
            res.status(400).json({message:fasle});
        }
        else
        {
            res.status(200).send(result);
        }
    })
})
app.listen(port,()=>{
    console.log("server is running");
});
```

Fig : Server.js

5.3) PERFORMANCE ANALYSIS :-

- Saved codes are stored in mongo dB database without any redundancy and latency.
- The time taken for code compilation depends on band width.
- The memory for link generation is handled efficiently.
- Users can successfully login through their google accounts.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1) CONCLUSION :-

By using our IDE, there is no need to save code in different folder or files, codes are stored in user profile itself. Users feel this more secured and comfortable. It is authenticated using Google Auth. Our product won't face any glitches or errors as it have simple architecture.

6.2) FUTURE WORK :-

- As of now our compiler can compile only 5 language, further we will increase the number of languages.
- Now our compiler can be used by competitive coders but in future we will can design in such a way that it can be used for web development.
- Present we are dependent on piston api but further we will create own api so that we can reduce the dependency.

REFERENCES

- **https://piston.readthedocs.io/en/latest/api-v2**
- **https://google-auth.readthedocs.io/en/master/**
- **https://github.com/engineer-man/piston**
- **https://www.npmjs.com/package/google-auth-library**
- **https://www.academia.edu/33264889**
- **https://www.researchgate.net/publication/292869015**