Hitarth Bhatt

Linkdin : https://www.linkedin.com/in/hitarth-bhatt/
Github : https://github.com/hitarthbhatt-31?tab=repositories
Email : hitarthbhatt12@gmail.com
Town, Country : Dungarpur, Rajasthan, India.

AOSSIE 2020 GSoC PROPOSAL

Project Title

Creating a fully functional Agora Vote IOS Application

Description

The idea behind this project is to create an IOS version of the Agora Vote application which leverages the existing Agora API to provide another way for Agora web users to use the platform using their IOS devices.

I have created a great UI design of the Agora Vote application and follow all the design principles. I also started the coding part of the application.
But there are some questions which answers can make huge benefits to the application.

1. Why will users use this application ?

As we know this question has lots of different answers but in my opinion, the most important factor is **User Experience (UX)** because **First Impression Is The Last Impression.**
Users always wants some major points in the application which are following :

- Great User Interface (UI).
- Fast In Performance.
- Easily Understandable.

-> For this application i have created a great UI Design which provides best User

Experience as a voting application and also started the coding part.
-> Another feature I have added is that in-app notification if a user gets a voting invitation.
-> Last but not least Swift UI provides best performance in IOS applications.

2. How will it be beneficial to the organization ?

It is beneficial to the open-source organization to earn profit from their software so they can make an improvement in their software as well as their organization.

There are different ways to earn profit from the application which point out below :

- **Provide full screen advertise in the application** :-
  - When a user completes its task afterwards a full screen ad pops on the view.
- Make subscription plan for users :-
  - Gives users limited access to adding candidates in election.
  - Providing long time period elections in paid plan.

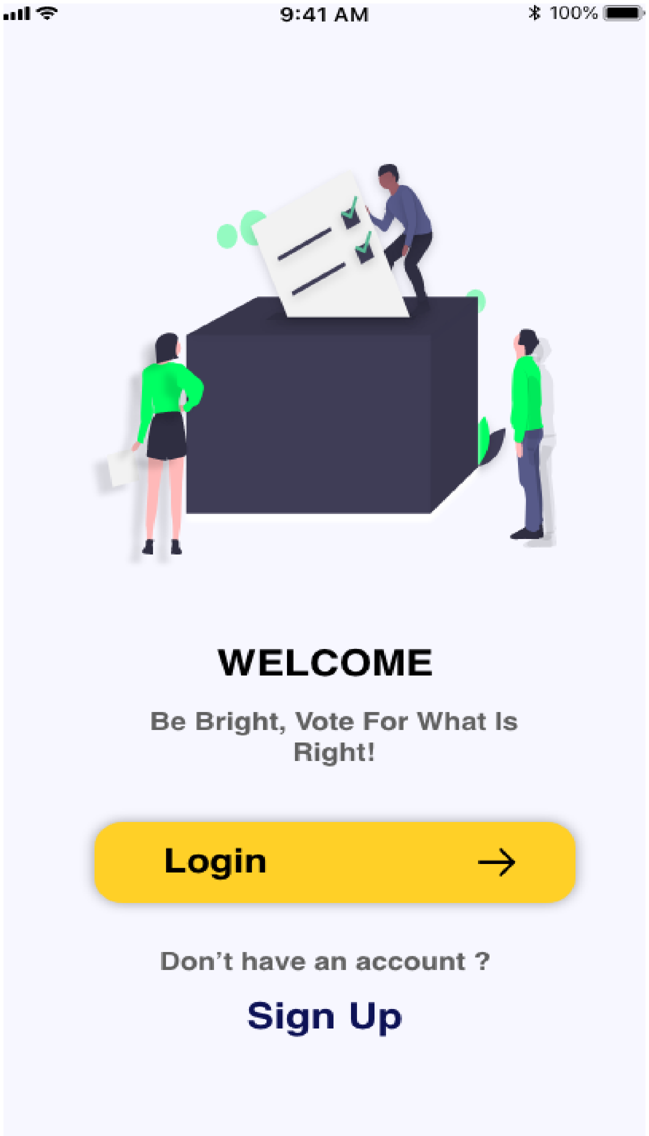
Application Development__

Now, for the IOS Agora Vote application I will use following frameworks for the development :

- Alamofire : It is an **HTTP networking library** written in Swift.
- SwiftyJSON : It makes it easy to deal with **JSON data** in Swift.
- Realm : Library for database to **save data** on the device.
- FacebookLogin : This open-source library allows you to integrate Facebook into your iOS app.

I already built an optimized welcome page and login page with autolayout functionality. The code and UI of these two pages shown below.

Welcome Page



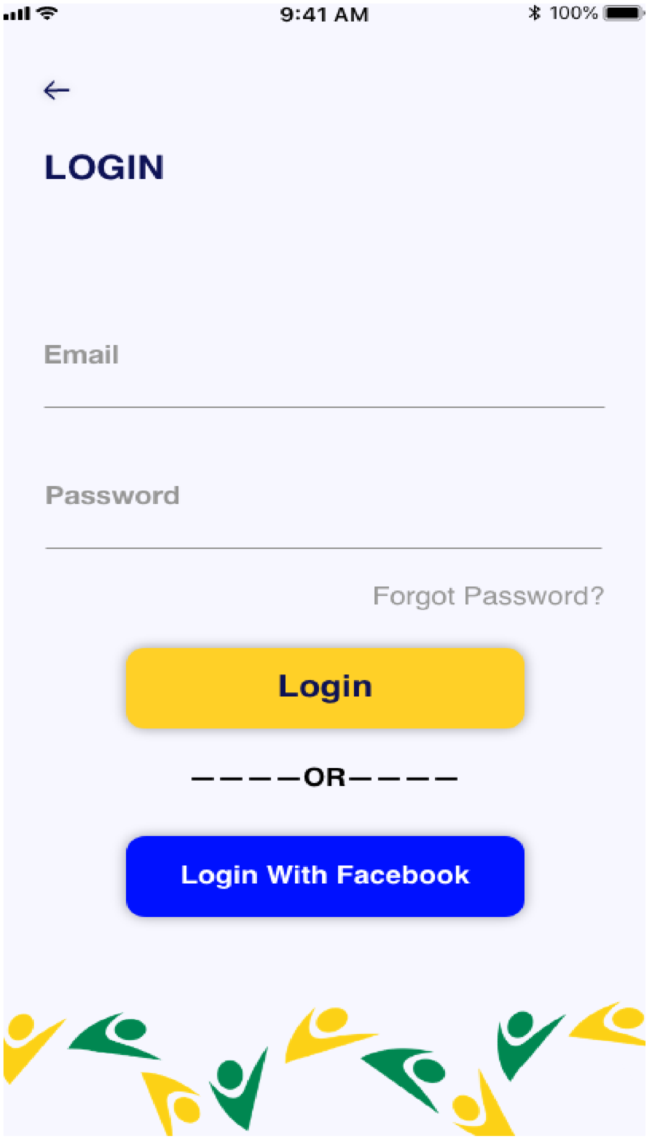
WELCOME

Be Bright, Vote For What Is Right!

Don’t have an account ?

```
struct WelcomeView: View {
    var body: some View {
        NavigationView {
            GeometryReader { geo in
                VStack {
                    //TODO:- Wlecome Page Image.

                    Image("Welcome_img")
                        .resizable()
                        .frame(width: (geo.size.width)/2, height: (geo.size.height)/3, alignment: .center)

                    //TODO:- Welcome Page Title
                    Text("Welcome")
                        .font(.largeTitle)
                        .fontWeight(.semibold)

                    Text("Be Bright, Vote For What Is Right!")
                        .lineLimit(2)
                        .frame(width: (geo.size.width)/2, height: (geo.size.height)/8, alignment: .center)
                        .multilineTextAlignment(.center)

                    //TODO:- Login Button.

                    NavigationLink(destination: LoginView(), label: {

                        HStack(spacing: (geo.size.width)/2) {
                            Text("Login")
                                .font(.system(size: (geo.size.width)/16))
                                .fontWeight(.semibold)


                            Image(systemName: "arrow.right")

                        }

                    }).foregroundColor(Color.init( colorLiteral(red: 0, green: 0, blue: 0, alpha: 1)))
                        .padding(18)
                        .background(Color.init( colorLiteral(red: 1, green: 0.8156862745, blue: 0.1529411765,
alpha: 1)))
                        .cornerRadius(15)
                        .shadow(color: Color.init( colorLiteral(red: 0.2683948864, green: 0.2176832431, blue:
0.03988369918, alpha: 0.3246348003)), radius: 6, x: 1, y: 1)


                    //TODO:- Sign Up Button.

                    Text("Don't have an account")
                        .lineLimit(1)
                        .font(.headline)
                        .frame(width: (geo.size.width)/2, height: (geo.size.height)/8, alignment: .center)
                        .multilineTextAlignment(.center)

                    NavigationLink(destination: SignUpView(), label: {
                        Text("Sign Up")
                            .font(.title)
                            .fontWeight(.semibold)
                            .foregroundColor(Color.init( colorLiteral(red: 0.05098039216, green: 0.07058823529,
blue: 0.337254902, alpha: 1)))

                    })

                    //VStack
                }

                //Geometry Reader
            }.background(Color.init( colorLiteral(red: 0.9675294757, green: 0.9676508307, blue: 0.997146666,
alpha: 1)))
                .edgesIgnoringSafeArea(.all)

            //Navigation View
        }

    }
}
```

Login Page



```
struct LoginView: View {

    @State var email: String = ""
    @State var password: String = ""
    @State var isAuthSuccess: Bool = false
    @Environment(\.presentationMode) var presentationMode: Binding<PresentationMode>

    var body: some View {
        GeometryReader { geo in

            VStack(spacing: (geo.size.height)/10) {

                Spacer()

                VStack(spacing: (geo.size.height)/20) {

                    TextField("Email", text: self.$email)
                        .padding(15)
                        .font(Font.system(size: 15, weight: .medium, design: .serif))
                        .overlay(RoundedRectangle(cornerRadius: 10).stroke(Color.gray, lineWidth: 1))
                        .padding([.leading,.trailing])

                    SecureField("Password", text: self.$password)
                        .padding(15)
                        .font(Font.system(size: 15, weight: .medium, design: .serif))
                        .overlay(RoundedRectangle(cornerRadius: 10).stroke(Color.gray, lineWidth: 1))
                        .padding([.leading,.trailing])

                    //VStack End
                }


                VStack(spacing: (geo.size.height)/25) {

                    //TODO:- Login Button.

                    NavigationLink(destination: HomeView(), isActive: self.$isAuthSuccess){
                        Button(action: {}, label: {

                            Text("Login")
                                .font(.system(size: (geo.size.width)/18))
                                .fontWeight(.semibold)
                                .foregroundColor(Color.init( colorLiteral(red: 0, green: 0, blue: 0, alpha: 1)))
                                .padding([.top,.bottom],(geo.size.height)/60)
                                .padding([.leading,.trailing],(geo.size.width)/4)
                                .background(Color.init( colorLiteral(red: 1, green: 0.8156862745, blue:
0.1529411765, alpha: 1)))
                                .cornerRadius(15)
                                .shadow(color: Color.init( colorLiteral(red: 0.2683948864, green: 0.2176832431,
blue: 0.03988369918, alpha: 0.3246348003)), radius: 6, x: 1, y: 1)
                        })
                    }

                    Text("- - - - OR - - - -")
                        .font(.headline)
                        .fontWeight(.semibold)


                    //TODO:- Facebook Login Button.
                    Button(action: {}, label: {

                        Text("Login With Facebook")
                            .font(.system(size: (geo.size.width)/22))
                            .fontWeight(.semibold)
                            .foregroundColor(Color.init( colorLiteral(red: 1, green: 1, blue: 1, alpha: 1)))
                            .padding([.top,.bottom],(geo.size.height)/55)
                            .padding([.leading,.trailing],(geo.size.width)/12)
                            .background(Color.init( colorLiteral(red: 0, green: 0.2, blue: 0.8, alpha: 1)))
                            .cornerRadius(15)
                            .shadow(color: Color.init( colorLiteral(red: 0.2683948864, green: 0.2176832431,
blue: 0.03988369918, alpha: 0.3246348003)), radius: 6, x: 1, y: 1)
                    })

                    //VStack End
                }
                Image("BottomImage")
                    .frame(width: (geo.size.width))

                //VStack End
            }.navigationBarItems(leading:

                    Button(action: {

                        self.presentationMode.wrappedValue.dismiss()
                    }, label: {
                        BackButton()
                            .frame(width: (geo.size.width)/12, height: (geo.size.height)/47, alignment: .center)
                    })
            )


            //Geometry Reader End
        }.background(Color.init( colorLiteral(red: 0.9675294757, green: 0.9676508307, blue: 0.997146666, alpha:
1)))
            .edgesIgnoringSafeArea(.all)
            .navigationBarTitle("Login", displayMode: .large)

    }
}
```

Customized Navigation Bar

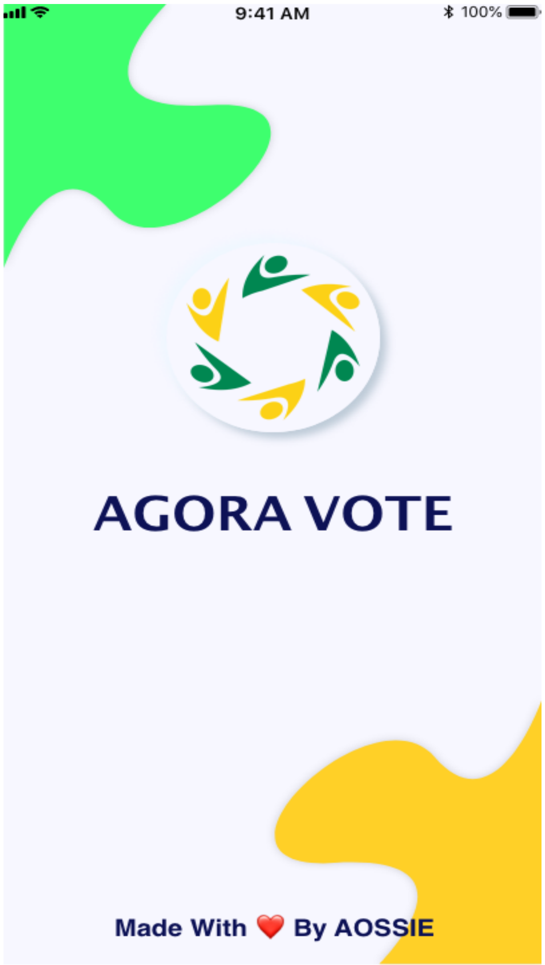
```
extension UINavigationController {

    override open func viewDidLoad() {
        super.viewDidLoad()

        let standeredAppearance = UINavigationBarAppearance()
        standeredAppearance.configureWithTransparentBackground()
        standeredAppearance.titleTextAttributes = [
            NSAttributedString.Key.foregroundColor : UIColor(red: 13.0/255.0, green: 18.0/255.0,
blue: 86.0/255.0, alpha: 1.0),
            NSAttributedString.Key.font : UIFont.preferredFont(forTextStyle: .title2)
        ]

        standeredAppearance.largeTitleTextAttributes = [
            NSAttributedString.Key.foregroundColor : UIColor(red: 13.0/255.0, green: 18.0/255.0,
blue: 86.0/255.0, alpha: 1.0)
        ]
        navigationBar.standardAppearance = standeredAppearance

    }

}
```

Customized Back Button

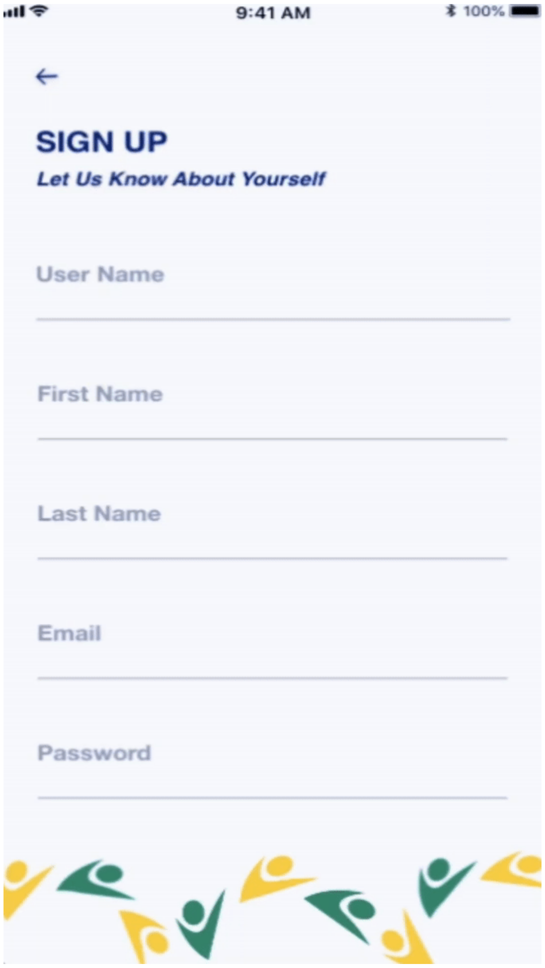
```
//TODO:- Back Button Setup.

struct BackButton: View {
    var body: some View {
        Image(systemName: "arrow.left")
            .resizable()
            .foregroundColor(Color.init( colorLiteral(red: 0.05098039216, green: 0.07058823529,
blue: 0.337254902, alpha: 1)))
            .padding(.leading,10)
    }
}
```

UI Design

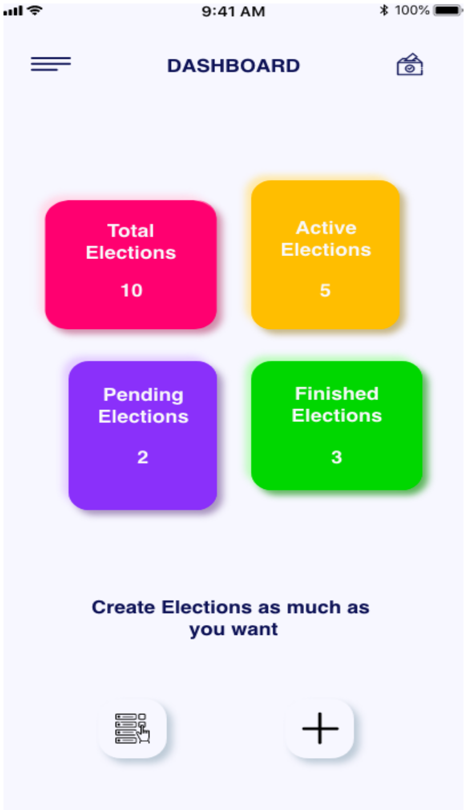
- **Launch Screen**



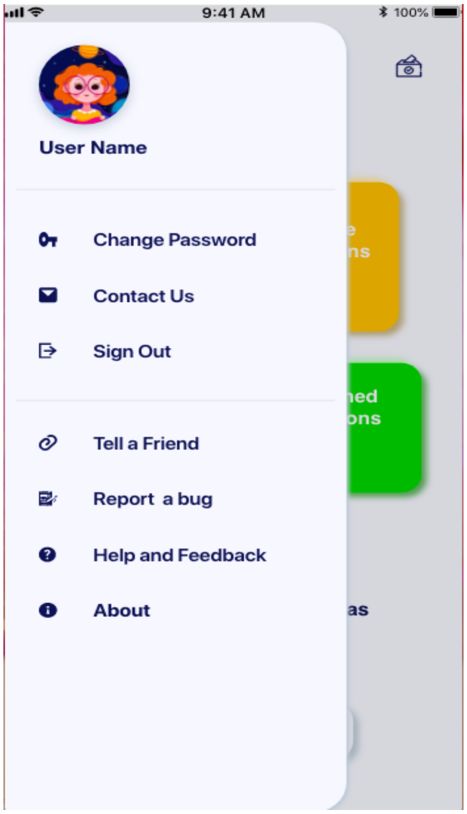
- **Sign Up View**



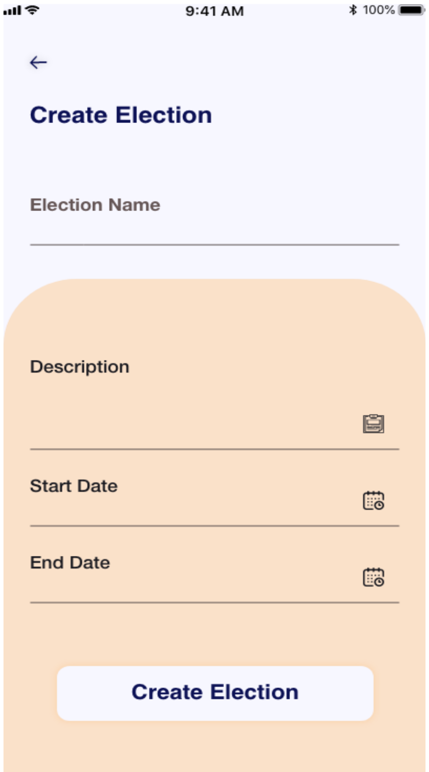
- Home View



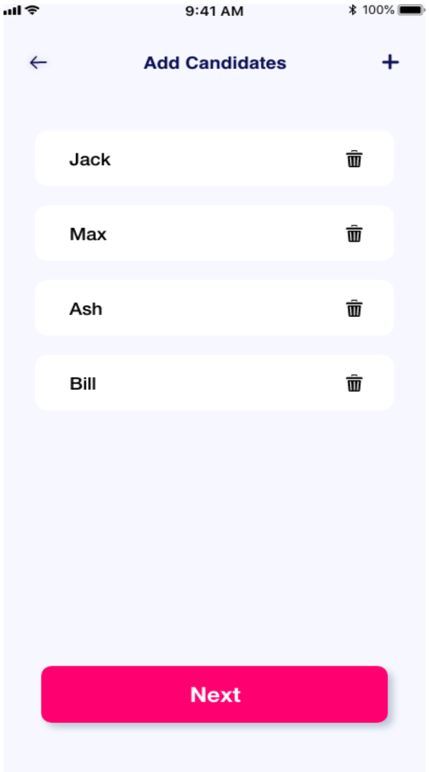
- Menu View



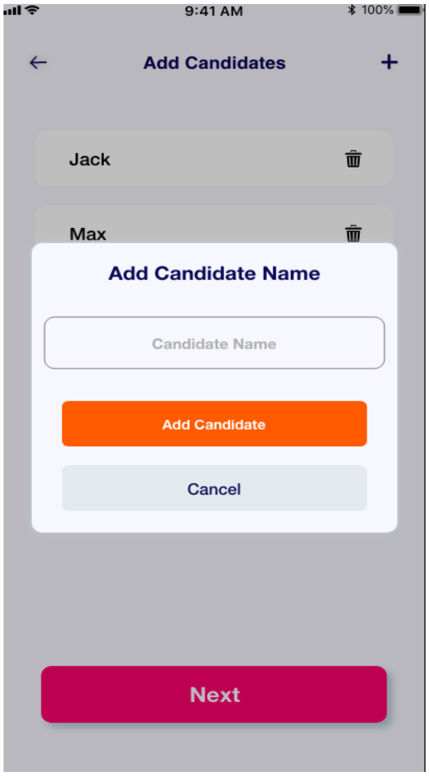
- Create Election



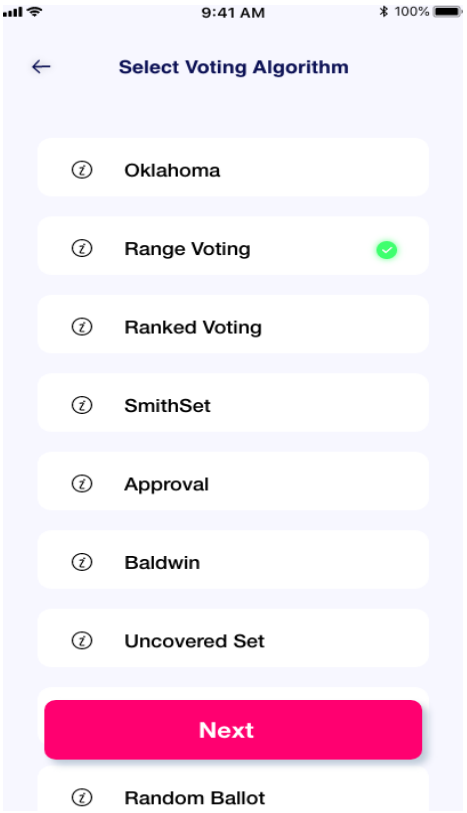
- Candidate Entry View



- Add Candidate View



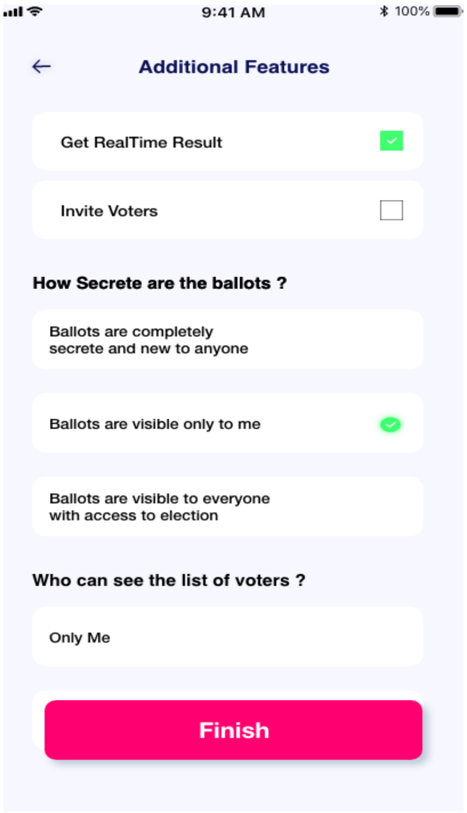
- Select Voting Algorithm View



Select Voting Algorithm

- Oklahoma
- Range Voting ✓
- Ranked Voting
- SmithSet
- Approval
- Baldwin
- Uncovered Set
- Random Ballot

- Additional Feature View



Additional Features

☑ Get RealTime Result

☐ Invite Voters

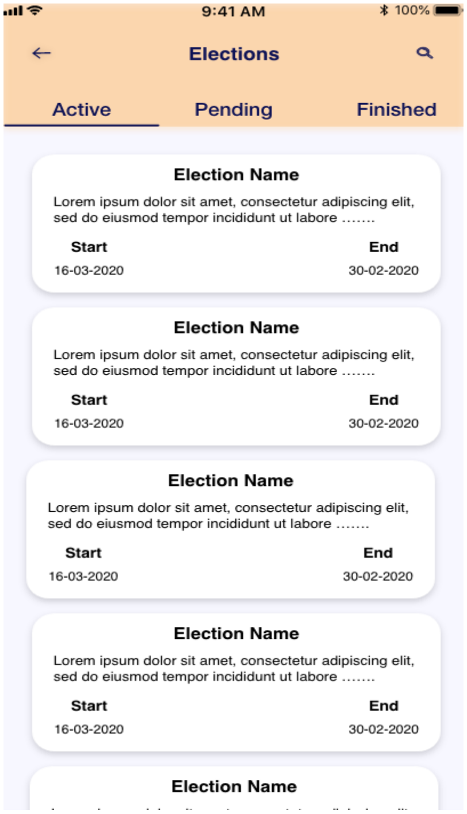
How Secrete are the ballots ?

- Ballots are completely secrete and new to anyone
- Ballots are visible only to me ✓
- Ballots are visible to everyone with access to election

Who can see the list of voters ?

Only Me

- Election View



Elections

Election Name

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore ....

Start	End
16-03-2020	30-02-2020

Election Name

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore ....

Start	End
16-03-2020	30-02-2020

Election Name

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore ....

Start	End
16-03-2020	30-02-2020

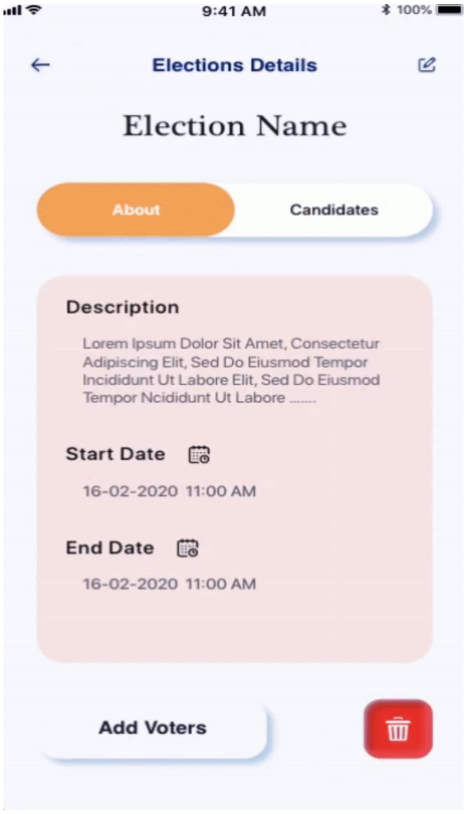
Election Name

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore ....

Start	End
16-03-2020	30-02-2020

Election Name

- Election Detail View



Elections Details

Election Name

Description

Lorem Ipsum Dolor Sit Amet, Consectetur Adipiscing Elit, Sed Do Eiusmod Tempor Incididunt Ut Labore Elit, Sed Do Eiusmod Tempor Ncididunt Ut Labore ....

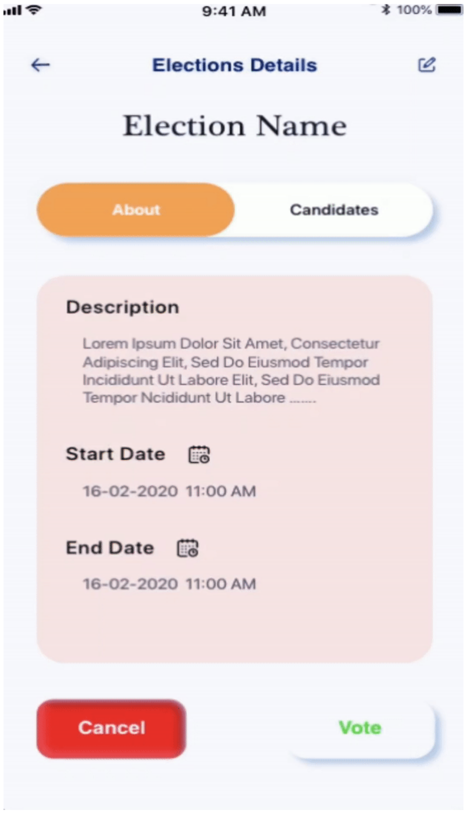
Start Date

16-02-2020 11:00 AM

End Date

16-02-2020 11:00 AM

- Voting Type A



9:41 AM 100%

Elections Details

Election Name

About Candidates

Description

Lorem Ipsum Dolor Sit Amet, Consectetur Adipiscing Elit, Sed Do Eiusmod Tempor Incididunt Ut Labore Elit, Sed Do Eiusmod Tempor Ncididunt Ut Labore ....

Start Date

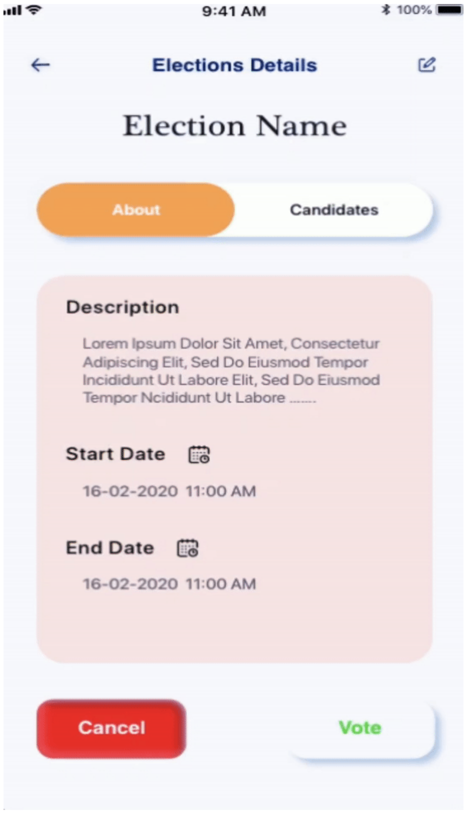
16-02-2020 11:00 AM

End Date

16-02-2020 11:00 AM

Cancel Vote

- Voting Type B



9:41 AM 100%

Elections Details

Election Name

About Candidates

Description

Lorem Ipsum Dolor Sit Amet, Consectetur Adipiscing Elit, Sed Do Eiusmod Tempor Incididunt Ut Labore Elit, Sed Do Eiusmod Tempor Ncididunt Ut Labore ....

Start Date

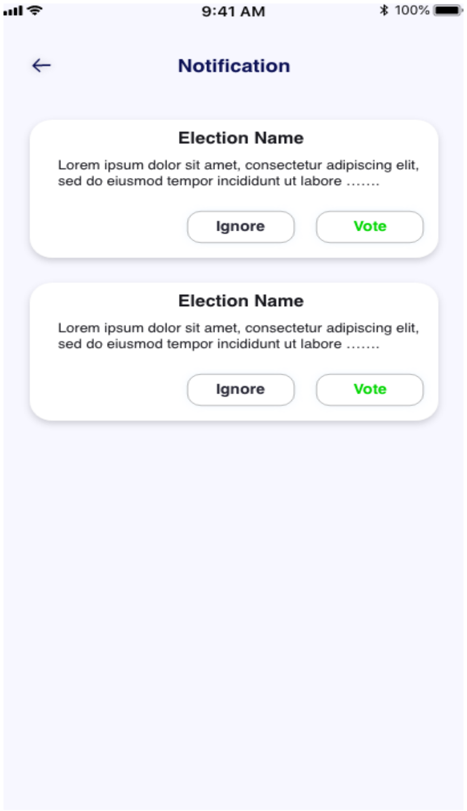
16-02-2020 11:00 AM

End Date

16-02-2020 11:00 AM

Cancel Vote

- Notification View



9:41 AM 100%

Notification

Election Name

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore ....

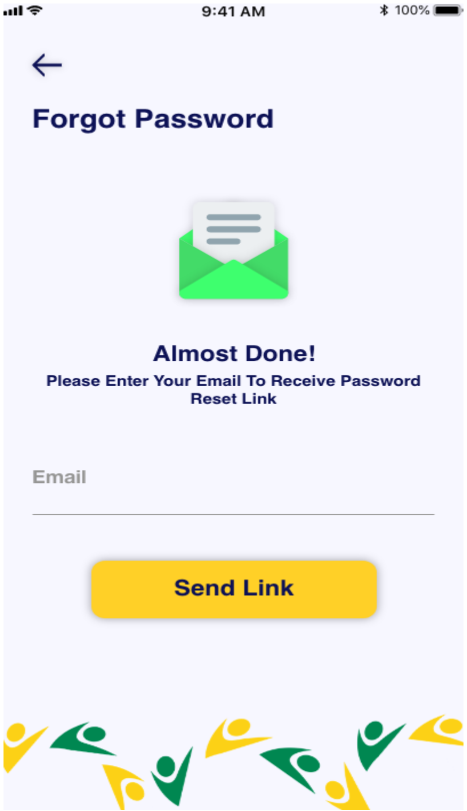
Ignore Vote

Election Name

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore ....

Ignore Vote

- Forgot Password View



9:41 AM 100%

Forgot Password

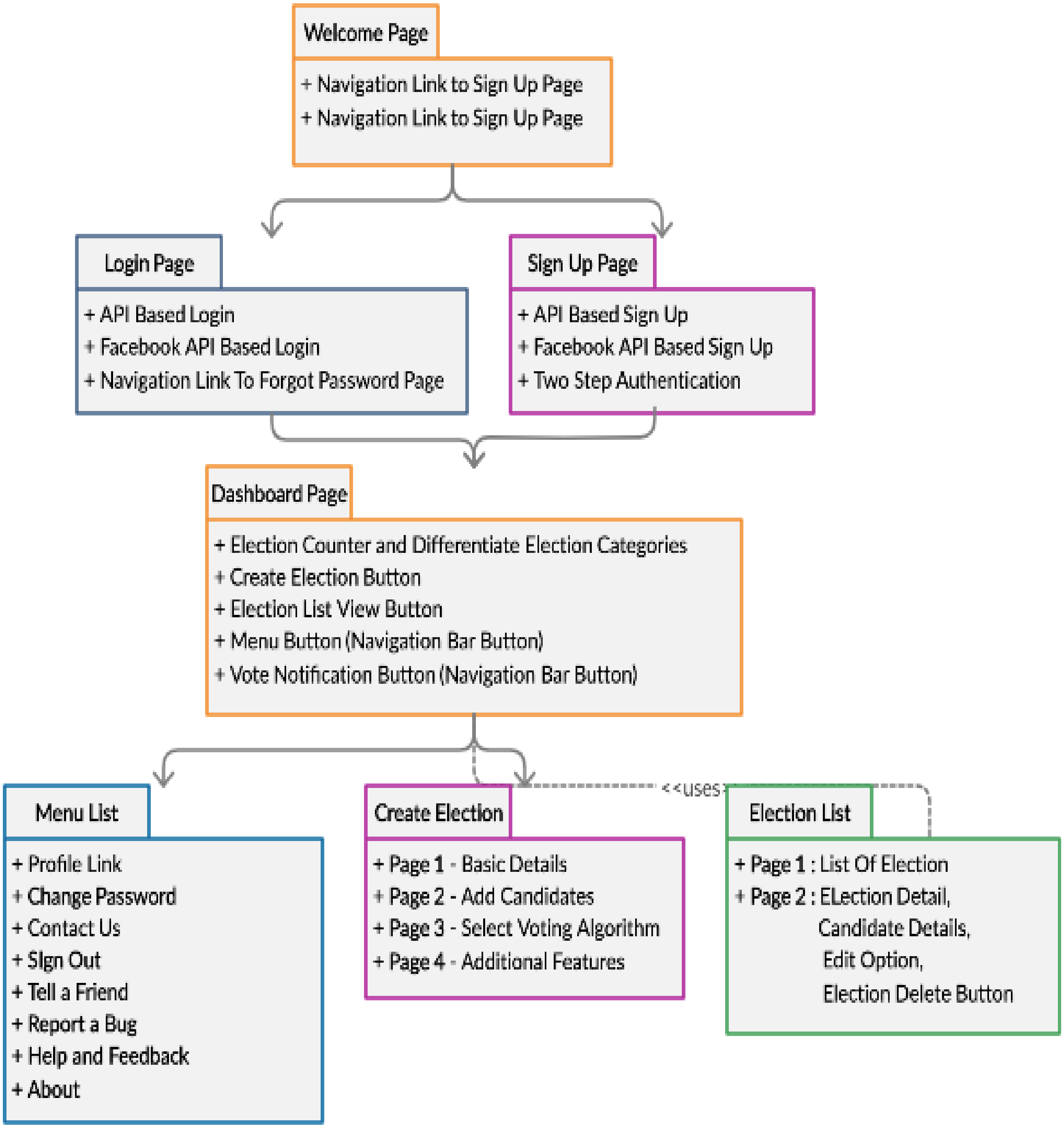
Almost Done!

Please Enter Your Email To Receive Password Reset Link

Email

Send Link

Application Architecture



Welcome Page

+ Navigation Link to Sign Up Page
+ Navigation Link to Sign Up Page

Login Page

+ API Based Login
+ Facebook API Based Login
+ Navigation Link To Forgot Password Page

Sign Up Page

+ API Based Sign Up
+ Facebook API Based Sign Up
+ Two Step Authentication

Dashboard Page

+ Election Counter and Differentiate Election Categories
+ Create Election Button
+ Election List View Button
+ Menu Button (Navigation Bar Button)
+ Vote Notification Button (Navigation Bar Button)

Menu List

+ Profile Link
+ Change Password
+ Contact Us
+ SIgn Out
+ Tell a Friend
+ Report a Bug
+ Help and Feedback
+ About

Create Election

+ Page 1 - Basic Details
+ Page 2 - Add Candidates
+ Page 3 - Select Voting Algorithm
+ Page 4 - Additional Features

Election List

+ Page 1 : List Of Election
+ Page 2 : ELection Detail,
  Candidate Details,
  Edit Option,
  Election Delete Button

Timeline

❖ Community Bonding Period (May 4 - June 1) :

- ➢ Enhance my understanding of Agora API and the Community.
- ➢ Discuss with the mentors and put together precisely components of my proposal and include the necessary changes.
- ➢ Learn how to implement dependency injection and tests for easy testing and verification of the work.
- ➢ Discuss with mentors for improving or make some changes in the UI.

❖ Week 1 (June 1 - June 8) :

- ➢ Finalize designs by discussing with mentors and community.
- ➢ Initialize Launch View, Welcome View, Login and Sign Up View in the Project.
- ➢ Add Alamofire and SwiftyJSON Framework in the project via Cocoapods.

❖ Week 2 (June 8 - june 15) :

- ➢ Use Agora Web API in the project for Login and Sign Up Activity with the help of framework.
- ➢ Create Home View with autolayout functionality.
- ➢ Add Side Menu in Home View and make a working Sign Out Button.

❖ Week 3 - 4 (June 15 - June 29) :

- ➢ Create Create Election, Add Candidate, Select voting algorithm and additional feature view in the project.
- ➢ Add Realm Database framework in the project via Cocoapods.
- ➢ Post the Created Election data to the server.
- ➢ Update UI (Home View) dynamically to show real time data changes.
- ➢ Clean up code and Documentation

❖ First Evaluation (June 29 - July 3)

❖ Week 5 - 6 (July 6 - July 20) :

- ➢ Create a Profile View Page with editing option and connect with the server.
- ➢ Create Election List and Election Detail page.
- ➢ Fetch data from the server and make editable election details.
- ➢ Make a temporary database via Realm to store the data in the device memory.
- ➢ Develop and write tests.

❖ Week 7 (July 20 - July 27) :

- ➢ Start working on all options which are shown in SideMenu.
- ➢ Testing of all features and views developed so far.
- ➢ Clean up code and documentation.

❖ Second Evaluation (July 27 - July 31)

❖ Week 8 - 9 (August 3 - August 17) :

- ➢ Start working on In-app voting functionality.
- ➢ Add different types of voting system in the UI (Eg : Shown above in UI section "Voting type A and Voting type B").
- ➢ Add Universal Linking feature in the project.
- ➢ Work on remaining works, If any.
- ➢ Add more features (as per discussion with mentors), If time permits.

❖ Week 10 (August 17 - August 24) :

- ➢ Finish unit and UI testing.
- ➢ Improve code quality and fix bugs, If any.
- ➢ Documentation of code.
- ➢ Submit final evaluation.

❖ Final Evaluation (August 24 - August 31)

Testing and Verification

During the coding period, I will push daily commits and send weekly pull requests. I will test thoroughly on a physical IOS device and make sure that the outcome is as it was planned to be. I will share screenshots and GIFs to show the actual working of the work done.

In the community Bonding Period, I will learn how to develop tests (Unit and UI tests) and write tests for all the features implemented and actively talk to the community.

References

1. Alamofire : Elegant HTTP networking in Swift.
   https://github.com/Alamofire/Alamofire

2. Realm Docs :
   https://realm.io/docs/swift/latest/

Experience and Education

I am currently in the third year, pursuing BTech, in IT at Arya College Of Engineering and I.T. (ACEIT). By the end of the GSoC period, I will have just started my fourth year of Bachelors.

In the field of development, I have been writing IOS applications in swift for the last 1 year. I have competed in hackathons like IncubateIND and RTU hackathon.

Currently I'm working on a project using cyber technology to prevent women and child violence and trafficking .

Good Fit

Open source has been a very interesting approach to learning new skills, working on projects in a like- minded community and challenging yourself to test what you know and learn what you don’t . Having developed tons of apps in IOS, I am confident enough to go through any codebase and perceive what’s going on.
I absolutely loved the way the community works. In the beginning I didn't know how to contribute to an open source organisation so I started cloning their projects and trying to understand their project.
FIrst open source organisation is Starscream and their project is based on Websocket So developers can easily understand and use it on their projects.

During the period of GSoC I will consistently work in a community over a period of time and being an active contributor, I think it only makes sense to continue contributing to AOSSIE in future and I am thrilled to put in my best! .

Other Commitments

- During the community Bonding period (May), I will take my finals So, in those days I will put in not more than **5-6 hours** daily to work on the proposed project.
- In the Coding period (June, July, August), I will totally commit to working on Agora IOS and put in **10-12 hours** on a daily basis.
- I have **NO** other obligation and I will work full time on the proposed project during the GSoC period.

Thank You