OhmniLabs User Authentication and Self Service Module

Project Author:

Cuong Nguyen chnguyen@wpi.edu

Project Advisor:

Professor Wilson Wong

Department of Computer Science

OhmniLabs Representative:

Tingxi Tan

Abstract

OhmniLabs, Inc. builds robotic platforms that power the future robot economy. One of the company products is Ohmni, a home robot that bring families closer through telepresence. Ohmni helps to enhance the user experience by having the robot be human height with a fluid moving neck so the call receiver can feel like he or she is talking to a real human. The project aims to build a user authentication module, a website for OhmniLabs new users to create their account so they can login to gain access to their Ohmni robots. The website should be reliable and easy for people who do not have experience using technology such as elderly people, who are the main customers of the company. The project is also creating a self-service module so the users can automatically reset their password in case that they forget it. The module will be integrated with the existing video call system of OhmniLabs.

Table of Contents

List of Figures

1.0 Introduction

Videotelephony is a type of technology which combines the reception and transmission of audio and video signals of different locations for communication between people in real-time. Telepresence is a reference to either a technology which goes beyond video where people can move around the room or physically manipulate objects, or a high-end video call. This creates the illusion of the participants being in the same room. The OhmniLabs, Inc vision is to connect people who are far apart from their families by creating a telepresence robot called Ohmni. Ohmni is a fresh approach to telepresence that is designed specifically for homes and families.

The project aims to build a user authentication module, which is a website for OhmniLabs new users to create their account, so they can log in to gain access to their Ohmni robots. The website will be reliable and easy for people who do not have experience using technology such as the elderly who are the main customers of the company. The project will create a self-service module so the users can automatically reset their password in case that they forget it. The module will be integrated with the existing video call system of OhmniLabs.

2.0 Background Research

Prior to designing and implementing any application, the author conducted research on technologies and framework that maybe useful, to determine the best technology stack with which to develop. This section covers the overview of OhmniLabs,Inc and the technologies explored for potential use through the project.

2.1 OhmniLabs,Inc

OhmniLabs, Inc was founded in 2015 and based in Sillicon Valley, CA. The co-founders, Jared Go, Tingxi Tan and entrepreneur Thuc Vu, are robotics experts from Carnegie Mellon and Stanford who believe that personal consumer robots can make a positive impact on people's everyday lives. They have extensive experience in AI, machine learning, multi-agent systems, industrial design, mechanical & electrical engineering and cloud infrastructure.

On April 12, 2017, OhmniLabs announced the launch of Ohmni, a home robot that lets families connect in a more natural and engaging way than any video chat. Ohmni is a fresh approach to telepresence that is designed specifically for homes and families. Ohmni makes it natural and effortless to join in and share experiences like cooking with mom, a family dinner, a movie night or a game day with dad -- across any distance. The team at OhmniLabs studied families around the world and found that these "shared experiences" were what people missed most since standard video chat was too cumbersome to enable these richer interactions.

Ohmni is designed from the ground up with simplicity in mind. To set up the robot, one can just unfold Ohmni, press the power button, and connect to a home Wi-Fi. Accessed is shared by inviting family and friends, by using an email address or Google or Facebook account. They can use the Google Chrome browser on any Mac/PC or Android device to connect to Ohmni from anywhere in the world. They can also interact freely by controlling Ohmni with the keys, trackpad, or touch controls.

Unlike other robotics and telepresence companies, OhmniLabs builds each robot locally in its facility in Santa Clara, CA using a scalable additive manufacturing process which they have tested and refined over the last two years. Every Ohmni is made to order and incorporates materials like bamboo and brushed aluminum to create a robot that users are proud to display in your home.

Ohmni was also created as an open and extensible platform through a cloud programming API known as OhmniAPI, which is designed to make programming robots as fast as developing web apps. Developers and businesses can build custom integrations on top of the full telepresence experience, the Ohmni software stack, or the low level hardware. Ohmni integrates with the Amazon Alexa API for home automation and general use when not in a call.(Tan, 2017)

2.2 Spam Prevention Methodology

Spam is an ongoing issue that requires a solution to protect the website from being abused by bots creating fake accounts. Therefore, we chose to use CAPTCHA and Email Confirmation as techniques to prevent spam bots.

2.2.1 CAPTCHA

Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) is a type of challenge-response test used in computing to tell whether a user is a human or not. The common test asks the user to retype the letter of a distorted image, sometimes with the addition of obscured letters or digits which appear on the screen.

reCAPTCHA

reCAPTCHA is a free CAPTCHA-live service from Google to establish that a computer user is a human or a bot in order to protect the website from spamming. They provide variations of tests like the image identification CAPTCHA, where the users have to choose multiple images that related to a topic, or the NoCAPTCHA reCAPTCHA where the users have to tick a box to confirm that they are not bots. We decided to use the NoCAPTCHA reCAPTCHA because it is a free service and highly effective in spam bot prevention.

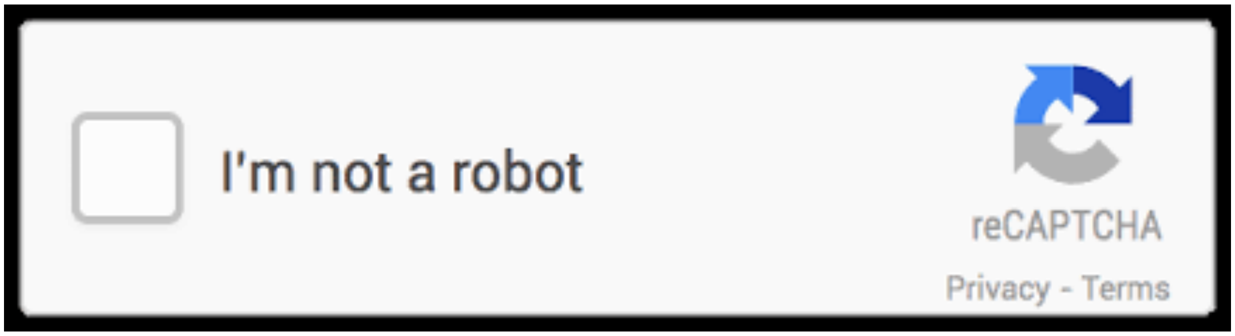


Figure 1 reCAPTCHA

3.0 Methodology

A software development methodology is as a framework that used to structure, plan and control the developing of an information system. The two main methodologies are the software development life cycle and the agile methodology. In this section, we will discuss these areas in greater detail and present the methodology that OhmniLabs is using.

3.1 Traditional Software Development Life Cycle

The Software Development Life Cycle (SDLC) has currently been the industry-standard approach to building high-quality software. The SDLC organizes development into sequential phases, with each phase forming the input for the next. This sequential nature is the reason why it is sometimes referred to as "a waterfall," and implies that the only motion is forward (TutorialsPoint, 2016). Consequently, it is expected that all work on any particular phase is completed before the next.

3.1.1 SDLC Stages

A usual SDLC consists of 6 stages from the following order: Planning – Analysis – Design – Implementation – Testing and Integration – Maintenance. The specific details for each phase varies among teams although these core steps are performed. Each phase is discussed in more detail below.

Stage one is the "Planning & Requirements Gathering". This step encompasses the activities pertinent to project initiation, such as team formation and decisions on the scope and objective(s) of the project. The team must engage in rigorous requirements gathering in order to make the most informed decisions at this step. The requirements gathering process can involve many steps such as literature reviews, surveys, and interviews (Shelly & Cashman, 2009).

Stage two is the "Analysis". In this phase, all the information gathered in the planning stage is organized. After that, the team uses each piece of information to develop the functional and nonfunctional requirements for the system. Questions like "Who is going to use the system?" and "How are they going to use the system?" are answered here.

The resulting set of functional and nonfunctional requirements will serve as the basis for developing the Analysis Model. Also called the Domain Model, this is a primarily visual document representing the entire system to be implemented, using real world terms (Shelly & Cashman, 2009). Since it is meant for all stakeholders to be able to understand, everyday language is used and not technical terms.

Stage three is the “Design” phase. In this stage, the team develops an Object Model for the system. An Object Model is a document detailing the exact design to be implemented, using computer-specific terms such as classes and objects. This document contains various diagrams including, but not limited to: class, sequence, and state diagrams. Last, and far from least, the team develops its Test Strategy in this step (Justin, 2013).

Stage four is the “Implementation” phase. At this stage, developers attempt to implement the features noted in their requirements documents using their preferred coding tools. This code-intensive stage is usually the longest period of the development cycle, possibly extending up to a few years (Shelly & Cashman, 2009).

Stage five is the “Testing and Integration” phase. During this stage, developers perform various automated and manual tests on the product, to ensure its adherence to the requirement specifications. Testing is done based on the predetermined test strategy, which is the team’s chosen approach to ensure methodical and thorough tests. After ensuring the integrity of the product, the team must decide how to deploy the application to the end user. This is the crux of the deployment step, and determines the team’s distribution medium (Justin, 2013)

Stage six is the “Maintenance” phase. After getting the product into the client’s hands, the team must ensure that it continues to work for a period of time agreed to by both the client and software development team. This step is also essential because software is almost guaranteed to encounter errors and the development team needs a plan on how to deal with the problems that arise (Shelly & Cashman, 2009).

The SDLC brings many advantages. The phases and guidelines of the SDLC are explicitly stated, allowing for the use of less experienced staff in software development. Furthermore, by being explicit with each step, the methodology promotes consistency across projects, allowing staff transfer between projects. Also due to the clearly structured phases, SDLC teams are able to better predict and meet deliverable deadlines; thorough planning is done for each step, accounting for contingencies.

However, it also has some disadvantages. For most projects, the SDLC is overly complicated, because it was designed to solve large, complex project issues. This is a hindrance to productivity, and can also result in a pile of redundant documentation. Additionally, the waterfall or “river of no return” model is infeasible, given that teams are highly unlikely to gather all requirements at the beginning of a project (TutorialsPoint, 2016). Furthermore, the customer is involved periodically, rather than being a full participant in the development process. This can lead to misinterpretations in requirements which could be left uncaught for too long. Sometimes, business changes invalidate the initial system design by the time of release, especially for long-term projects. With frequent involvement from clients, the problem of development going off-track could be circumvented at an early, relatively inexpensive stage (Putnam-Majarian & Putnam, 2015).

3.2 Agile Methodologies

The Agile approach was developed to counteract the limitations of the traditional waterfall methodology, particularly those brought on by over-complexity and lack of flexibility of the process. Paramount was the loss in productivity to practically meaningless tasks, such as heavy documentation. Furthermore, the lack of frequent stakeholder input often led to disparities between stakeholder expectation and software reality. In searching for solutions to the problems of the SDLC, developers adapted the manufacturing industry's "Lean Manufacturing techniques" to form the Agile software development techniques. Where being agile denotes "flexibility and responsiveness within a well-defined context", with a strong emphasis on results and the business value of each step in the process("What is agile? What is Scrum?," 2013).

We will focus on the Kanban approaches, which is the one OhmniLabs is using at the time of this report. However, before we go into the details of Kanban, we will first examine the benefit of agile methodologies.

3.2.1 Benefits of Agile

First of all is the benefit of stakeholder involvement. A stakeholder refers to any person who will be involved with the application, whether during development or after deployment. The Agile approach strongly emphasizes constant collaboration between all stakeholders, particularly between clients and project teams, to ensure that the product meets expectation.

Next, Agile methodology comes with high flexibility. As mentioned above, stakeholder input is key at each step of the agile approach, i.e. each iteration. As such, clients are able to communicate any feedback on development over short time periods. This allows the team to adapt quickly to minute changes, rather than present a totally unsuitable application after tedious development.

Timely delivery schedule is the next benefit of Agile. This pertains more to the fixed-length cycles of Scrum, where development is limited to a fixed amount of time, usually 2 weeks. Consequently, feature development is done in sizeable chunks, which fit in the development schedule as enforced by the project manager.

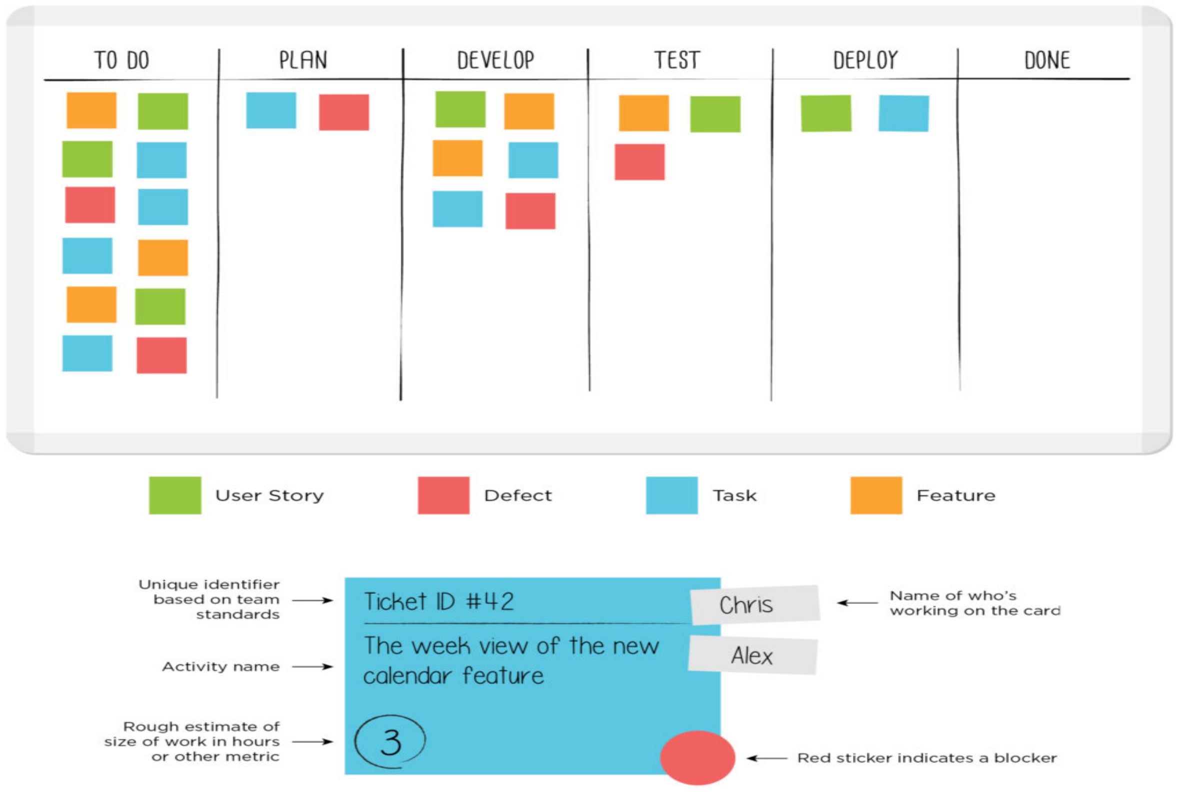
Agile methodologies also help to predict costs. With given fixed-length iterations, it is easy to estimate how much a sprint is going to cost in terms of time or other required resources. This in turn enables the team to predict any changes that may occur to the schedule and prepare accordingly. In addition, by making sure that every step taken is crucial to the development process, teams can avoid any worthless or redundant tasks. This focus ultimately increases team efficiency and productivity.

3.2.2 Kanban

Kanban is another application of agile methodologies that has been popular recently. It was adapted from the "just-in-time" (JIT) manufacturing model at Toyota motors, which was in turn borrowed from the supermarket industry. The model is simple; stock just enough goods to meet customer demand. This prevents inventory-related issues by minimizing excesses, while serving the needs of customers. To understand how team, adhere to Kanban, it is essential to discuss Kanban boards, an integral part of the approach(Atlassian, 2016).

3.2.3 Kanban Board

This is a physical or digital tool used to visualize and optimize the team's workflow. A basic Kanban board consists of a 3-step workflow categorized under "To Do", "In Progress" and "Done". However, variation can be made in order to match the needs of the team. For our sample board, we can have some additional columns like "Plan", "Test", and "Deploy".



TO DO
PLAN
DEVELOP
TEST
DEPLOY
DONE
User Story
Defect
Task
Feature
Unique identifier based on team standards
Activity name
Rough estimate of size of work in hours or other metric
Ticket ID #42
The week view of the new calendar feature
3
Chris
Alex
Name of who's working on the card
Red sticker indicates a blocker

Figure 2 Kanban Board (LeanKit)

3.2.4 Benefits of Kanban

Kanban can bring extremely flexibility in planning. Developers simply pull the next item off the product backlog and implement it, paying no attention to anything else. As such, product owners are able to freely organize the backlog in a way that makes practical sense for the team, without consequence to the team's work.

It also makes the cycle time shorter. The amount of it takes a unit of work to go through the entire workflow process - from start to finish - is known as its cycle time. This is one of several key metrics for Kanban teams, enabling teams accurately predict future delivery times. As a preventive measure, Kanban teams employ work-in-progress (WIP) limits to different stages in the workflow process. For example, a team may place a limit of 2 items under "code review", signifying that before any work items can be added, the current ones must be dealt with. If it becomes necessary, work in other steps can be halted, to get all hands on deck to deal with a bottleneck, cutting down on overall cycle time.

Continuous delivery (CD) refers to the practice of releasing products to customers regularly and frequently. This is only possible by a highly flexible team that can quickly respond to customer needs, making kanban with its JIT principles the perfect solution.

4.0 Software Development Environment

In this section, we will discuss our development environment at OhmniLabs. These tools and processes were selected by the company to provide a speedy development.

4.1 Project Management Software

Project management software is software used for project planning, scheduling, resource allocation and change management. It allows project managers (PMs), stakeholders and users to control costs, and manage budgeting, improve quality management, track documentation and also may be used as an administration system. Project management software is also often used for collaboration and communication between project stakeholders.(Techopedia, 2017b)

4.1.1 Trello

At OhmniLabs, Trello is used as the main project management software. Trello is another web-based tool which is primarily built for team collaboration, but it has excellent project management capabilities. It features an easy-to-use card system, where tasks are placed on individual cards and can then be assigned to team members (Duffy, 2016). These cards have associated boards, which are designed to represent projects but can be organized as desired due to its vague specification. Trello allows team members to see what tasks are being worked on and the status of those tasks by organizing cards into lists. Typical categories include: Planned, In-Progress, Done, and Review. This helps the team keep up-to-date on how much functionality has been completed within a given iteration and allows it to more effectively plan the next iteration. Although seemingly basic, Trello offers a rich, customizable visual interface. This means that colors and shapes are designed to a team or individual's specifications. This enables the use of color coding as well as other mechanisms to improve team productivity (Duffy, 2016). Secondly, due to its inherent cloud-based nature, teams have access to their boards at any time from a variety of devices. It is available on PC via a web app, and can be used on several mobile devices including Android and iOS devices. Many users also find Trello extremely easy to use with its intuitive interface. This ease is further enhanced by the integration of third-party applications through Zapier. Most importantly in our case, Trello is completely free to use, although special features are only accessible with a premium membership.

4.2 Team CollaborationSoftware

Collaboration software enables the sharing, processing and management of files, documents and other data types among several users and/or systems. This type of software allows two or more remote users to jointly work on a task or project. Collaboration software is primarily designed to enhance productivity within a group of individuals and, more specifically, within organizations. This is achieved through the coordinated tasks processing and management capabilities provided by this type of program. With collaboration software, users each create a workspace and add data and/or workflows to it. The created workspace is viewable and accessible by all other users, regardless of their physical location, and can be given access to the workspace by its primary user. Any changes made to the data or files are synced across all users by the collaboration software, ensuring that everyone has the most updated version of an ongoing project (Techopedia, 2017a). In the next section, we will discuss our communication software, Slack, in more detail.

4.2.1 Slack

Slack is our main communication software for development and meeting. Slack is a real-time team communication tool. It organizes conversations into channels, allowing teams to create multiple public and private ones for different purposes (LaGreca, 2018). Slack channels facilitate a great user experience. Teams are unrestricted in how many channels can be created. Users are free to participate in any number of public channels, and private channels to which they are granted access. This creates an intuitive way to organize communication and teams can easily develop an efficient system of use. Users are able to share code snippets or entire source files via a built-in code viewer. In addition, members can share files via a simple drag-and-drop. Slack provides extensive search, even to the contents of the shared documents. Coupling these sharing capabilities with features such as comments and reactions to message items give teams a fully comprehensive communication tool. Moreover, Slack offers several integrations essential to software teams such as Trello and GitHub, among others. All of these amazing features are available to use for free for a team of any size.(LaGreca)

4.3 Repositories

A repository is a central file storage location including a version control system to store multiple versions of files. At OhmniLabs, we used GitLab as our main repository, which we will discuss more details in this section.

4.3.1 GitLab

GitLab is the main repository that we use at OhmniLabs. GitLab is an online Git repository manager with a wiki, issue tracking, Continuous Intergration (CI) and Continuous Deployment(CD) and an easier way to manage git repositories on a centralized server. GitLab gives you complete control over your repositories or projects and allows you to decide whether they are public or private for free (GitLab). GitLab has a convenient user interface which enables user to access everything from one screen. It also has a variety of authentication levels which allows user to give people access beyond read/write level. Protected branch is another key feature in GitLab to keep code safe by allowing users to set higher permissions on a project so only certain people are able to push or delete that branch. Using the code snippet feature, a user can share small piece of code from a project without sharing a whole project.

4.4 Database

A database is a data structure that stores organized information. Most databases contain multiple tables, which may include several different fields. OhmniLabs uses Redis as their database.

Redis (REmote DIctionary Server) is an in-memory, key-value database, commonly referred to as a data structure server. One of the key differences between Redis and other key-value databases is Redis's ability to store and manipulate high-level data types. These data types are fundamental data structures (lists, maps, sets, and sorted sets) with which most developers are familiar. Redis's exceptional performance, simplicity, and atomic manipulation of data structures lends itself to solving problems that are difficult or perform poorly when implemented with traditional relational databases.(Shon, 2014).

There are many benefits that Redis brings to the users. First, "In-memory data structures" which means that Redis allows users to store keys that map to various data types. The fundamental data type is a String, which can be text or binary data with a size of up to 512MB. Redis also support Lists of Strings in the order they were added; Sets of unordered Strings; Sorted Sets ordered by a score; Hashes which store a list of fields and values; and HyperLog to count the unique items in a data set. Nearly any type of data can be stored in memory using Redis. Secondly, Redis brings versatility and ease-of-use by providing a number of tools that make development and operations faster and easier. Pub/Sub, to publish messages to channels, which are delivered to subscribers are great for chat and messaging systems.TTL keys can have a determined Time To Live, after which they delete themselves which is useful to avoid filling the database with unneeded data. Atomic counters ensure that race conditions do not create inconsistent results. Redis also provides Lua, a powerful but lightweight scripting language. Moreover, replication and persistence is a key feature of Redis. Redis employs a master-slave architecture and supports asynchronous replication where data can be replicated to multiple slave servers. This can provide both improved read performance, (as requests can be split among the servers, and recovery when the primary server experiences an outage. To provide durability, Redis supports both point-in-time Snapshots, copying the Redis data set to disk, and creating an Append Only File (AOF) to store each data change to disk as it is written. Both methods allow rapid restoration of Redis data in the event of an outage. Lastly, Redis supports many languages including Java, Python, C, C++, and Node.js (Shon, 2014).

4.5 Integrated Development Environment

For the Integrated Development Environment (IDE), there are two different ones for the Linux staging machine and the local machine: Vim for the Linux machine and WebStorm for the local machine.

4.5.1 Vim

Vim is the editor that is used in development at the Linux staging machine at OhmniLabs. Vim (Vi Improved) is a clone of vi text editor program for Unix. Vim is designed for use both from a command-line interface and as a standalone application in a graphical user interface. Vim can be customized easily to match the user needs and has a lot of plugins that can extend or add more functionality. Vim is easy to install on any Linux machine and does not take a lot of space in memory (Ilic, 2014).

4.5.2 WebStorm

WebStorm is the editor that is used in development on the local machine. WebStorm is a high-end IDE for web development project from Jetbean (Choudhry, 2016). In addition to supporting JavaScript, HTML, CSS, WebStorm also supports many popular frameworks like Angular, React, Node.js, Cordova, or React Native. Compared with other JavaScipt IDEs, WebStorm has many features that make it "smarter". First, WebStorm provides advanced coding assistance for several web frameworks and build-in inspection for checking code error, so the developer can write high quality code and fix all errors quickly. Webstorm also has Git version control built-in so developers can commit files, review changes and resolve conflicts with a visual diff/merge tool in the IDE (Choudhry, 2016).

5.0 Software Requirement

5.1 Functional and Nonfunctional Requirements

Functional requirements are "what the system should do" and nonfunctional requirements "describe how the system works" (Eriksson, 2012). The requirements for this project are:

- Integrate third party authentication like Facebook and Google sign in to OhmniLabs existing sign in methods.
- Develop account recovering and spam checking capabilities for user logins.
- Develop onboarding workflows for when a user first signs up on the web app.

5.1.1 Functional Requirements

Functional requirement are descriptions of functions of a system or its component. These requirements specify the results of a system.

Account Creation

A user can create a new account by providing full name, email address and password. After signing up, the application will send a confirmation email to the email address that the user provided. When the user has successfully confirmed the account, the application will add that account to the database. Afterwards, users can log in to the website with their usernames and password.

Account Authorization

Users will be required to log in to the website when they start it. Users can log in with OhmniLabs accounts or they can choose to log in with their Google or Facebook accounts. When logging in with a third party application like Google or Facebook for the first time, the website will ask users to log in to their Google or Facebook accounts. Afterwards, their public email address from the third party accounts will be added to the database.

Account Retrieval

When a user wants to reset an account password that they forgot, the user has to enter an email address and the system will check if that email exists in the database. If it exists, an email will be sent to that address with the email message to reset their account password.

5.1.2 Nonfunctional Requirements

Nonfunctional requirements specify criteria that can be used to judge the operation of a system rather than specific behaviors. These requirements place restrictions on the product being developed and the development process, and specify constraints that the product must meet. The nonfunctional requirements of this project are discussed in more details below.

Security

We enforce users to create good passwords by requiring at least eight characters, at least one number and at least one special character. The system will also enable reCAPTCHA, if the user enters their password incorrectly three times, to check if this is a human or a bot. The user can request to reset their password if he or she is unable to login. The system will send a link to the user's email address so the password can be reset.

Usability

The login and sign-up account is to be designed to be as easy as possible for the user. The design needs to be intuitive and simple so a user with little background about computers can create an account and log in.

Performance

The website needs to be available and downtime minimized so users can log in and sign up whenever they want.

5.2 Epics

In our project, we have the two user roles:

1. Administrator
2. OhmniLabs User

5.3 User Stories

a. As an OhmniLabs User, I would like to be able to create a new account in order to enter and use OhmniLabs's service.

b. As an Administrator, I would like to be able to unlock user accounts when asked to satisfy user requests.

c. As an OhmniLabs User, I would like to be able to reset my account password in order to enter and use OhmniLabs's service.

d. As an OhmniLabs User, I would like to be able to sign in with a Facebook or Google account in order to enter and use OhmniLabs service.

e. As an Administrator, I would like to be able to delete a user account when my user asks to satisfy their requests.

f. As an Administrator, I would like to be able to add a new user account when asked to satisfy user requests.

g. As an Administrator, I would like to be able to link a user account with an Ohmni bot when asked to satisfy user requests..

h. As an Administrator, I would like to be able to unlink a user account from an Ohmni bot when asked to satisfy user requests..

5.4 Scenarios

A scenario is a scene that illustrates some fictional interaction with a proposed system. It is a tool to describe a specific use of a proposed system. We will present our scenarios in this section.

Administrator

Tingxi is the administrator at OhmniLabs. He searches through the database to find if a user account exists. The search produces the user account data. Since the user requests to delete his account, Tingxi selects that account and deletes it from the database.

OhmniLabs User

Cuong needs an OhmniLabs account to make a video call to his Ohmni bot. Cuong goes to the OhmniLabs website and clicks the sign-up button. The website redirects to a sign-up form which requires an email address and password. Cuong enters his account information and presses submit. Cuong gets a confirmation email message from OhmniLabs to confirm his email address. Cuong clicks the link in the confirmation email. The OhmniLabs website will verify and redirect Cuong to the main page. Cuong can now log in to his account after verification.

5.5 Use Cases

A use case is a generalized description of a set of interactions between the system and one or more actors, where an actor is either a user or another system. It describes a process and its steps in detail, and may be worded in terms of a formal model. We will now present the use cases from our analysis.

Use Case #1

i. Name: Create new OhmniLabs account

j. Participating Actor: OhmniLabs User

k. Entry Conditions:

l. The user does not existed in the database

m. Exit Conditions:

n. The user account exists in the database

o. Flow of Events:

p. OhmniLabs User goes to the sign-up page of OhmniLabs website

q. OhmniLabs provides a form to capture the new user information like full name, email address and password.

r. OhmniLabs User fills the form and clicks the button to register.

s. OhmniLabs sends a confirmation email to the user's email address.

t. OhmniLabs User verifies his/her account.

u. OhmniLabs added the new account to the database.

v. Alternate Flows of Events

w. OhmniLabs User cancels at an intermediate step.

x. OhmniLabs does not get the verify back from the user and does not create a new account.

Use Case #2

1. Name: Sign in with OhmniLabs account
2. Participating Actor: OhmniLabs User
3. Entry Conditions:
    a. The user exists in the database

4. Exit Conditions:
    a. The user can log in to his/her account.
5. Flow of Events:
    a. OhmniLabs User goes to the sign in page of OhmniLabs website
    b. OhmniLabs provides a submission form so the user can sign in with his/her email address and password.
    c. OhmniLabs checks if the user enters a correct email-password.
    d. OhmniLabs authorizes and the user is signed in.
6. Alternate Flows of Events
    a. OhmniLabs User cancels at an intermediate step.
    b. OhmniLabs User enters an incorrect email/password.

Use Case #3

1. Name: Forgot OhmniLabs account password
2. Participating Actor: OhmniLabs User
3. Entry Conditions:
    a. The user exists in the database
4. Exit Conditions:
    a. The user gets an email with instructions to reset their account password.
5. Flow of Events:
    a. OhmniLabs User goes to the sign in page of OhmniLabs website
    b. OhmniLabs User presses the Forgot Password button.
    c. OhmniLabs User enters their email address.
    d. OhmniLabs checks if the email exists in the database.

    e. OhmniLabs sends an email with instruction to that email address.

6. Alternate Flows of Events
    a. OhmniLabs User cancels at an intermediate step.
    b. OhmniLabs User enters an incorrect email.

Use Case #4

1. Name: Reset OhmniLabs account password
2. Participating Actor: OhmniLabs User
3. Entry Conditions:
    a. The user exists in the database
4. Exit Conditions:
    a. The user can set a new password for heir OhmniLabs account.
5. Flow of Events:
    a. OhmniLabs User gets an email from OhmniLabs after pressing the Forgot Password button and entering their email address.
    b. OhmniLabs User clicks the link in the email.
    c. OhmniLabs User goes to the reset password page.
    d. OhmniLabs User enters a new password and reenters it a second time.
    e. OhmniLabs checks if both of the passwords match and sets it to the OhmniLabs User new password.
6. Alternate Flows of Events
    a. OhmniLabs User cancels at an intermediate step.
    b. OhmniLabs User reentered password does not match the new password.

Use Case #5

1. Name: Lock Account
2. Participating Actor: OhmniLabs User
3. Entry Conditions:
    a. The user exists in the database
4. Exit Conditions:
    a. The user account can log in to his/her account.
5. Flow of Events:
    a. OhmniLabs User goes to the sign-up page of OhmniLabs website
    b. OhmniLabs User enters password 3 times incorrectly.
    c. OhmniLabs locks the user account.
    d. OhmniLabs User goes to the forgot password page.
    e. OhmniLabs sends a reset password link to the user.
6. Alternate Flows of Events
    a. OhmniLabs User cancels at an intermediate step.
    b. OhmniLabs does not get the verify back from the user and does not reset the password and unlock the account.

Use Case #6

1. Name: Delete OhmniLabs account
2. Participating Actor: OhmniLabs Adminstrator
3. Entry Conditions:
    a. The user account exists in the database
4. Exit Conditions:
    a. The user account does not exist in the database

5. Flow of Events:
    a. OhmniLabs User goes to the sign-up page of the OhmniLabs website
    b. OhmniLabs provides a form to capture the new user information such as full name, email address and password.
    c. OhmniLabs User fills the form and clicks the button to register.
    d. OhmniLabs sends a confirmation email to the user’s email address.
    e. OhmniLabs User verifies his/her account.
    f. OhmniLabs adds the new account to the database.
6. Alternate Flows of Events
    a. OhmniLabs User cancels at an intermediate step.
    b. OhmniLabs does not get verification back from the user and does not create a new account.

Use case diagram

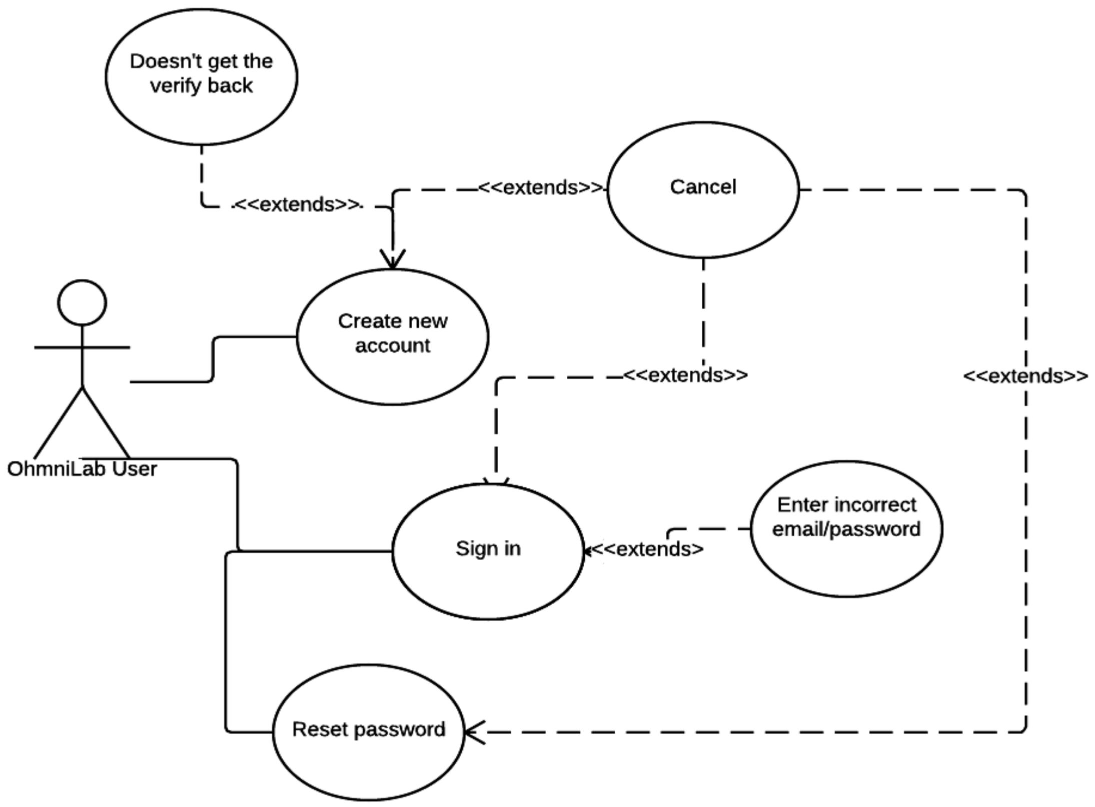


Figure 3 Login Website Use Case Diagram

5.6 Interface Mockups

5.6.1 *Account Creation*

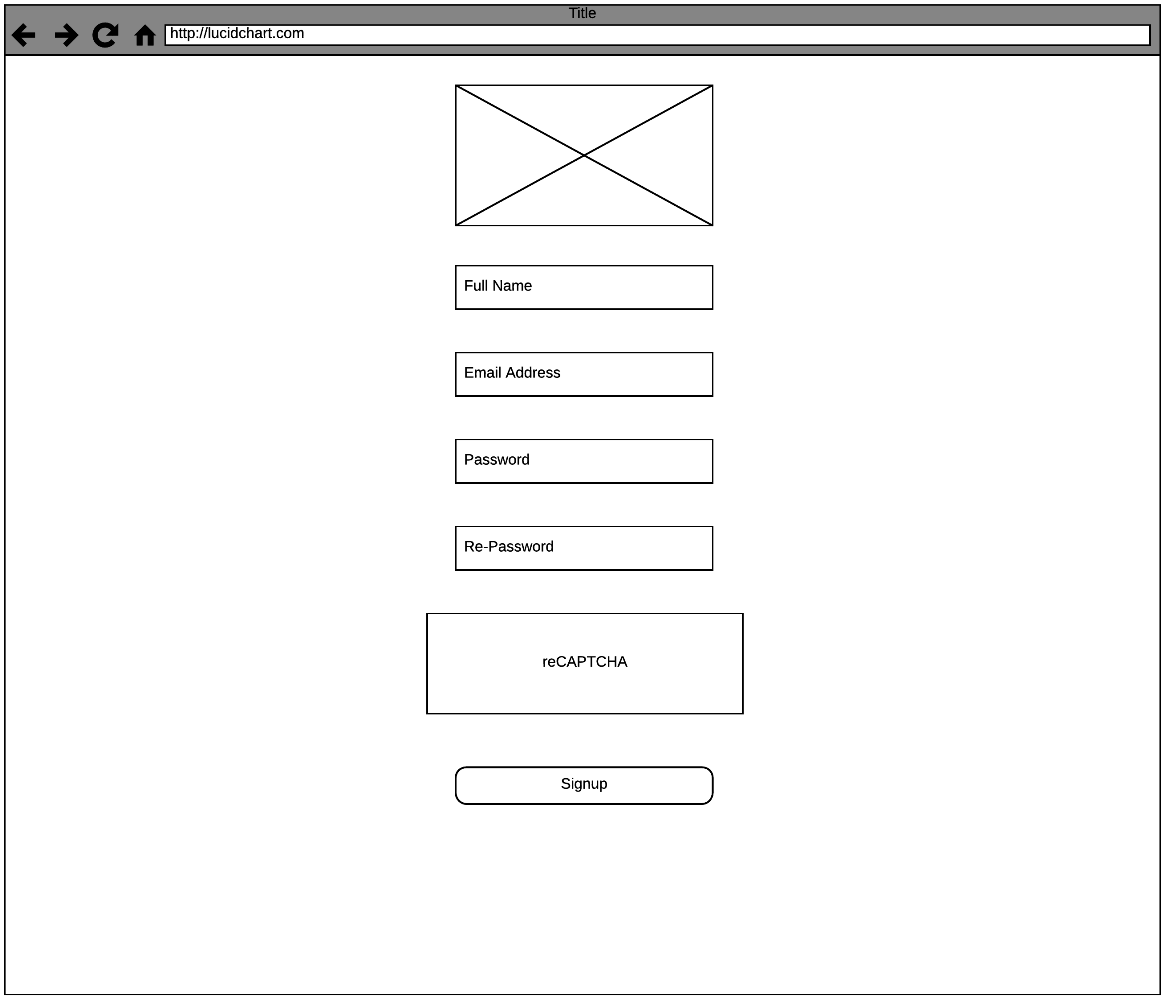


Figure 4 Account Creation Mockup

This is the screen that users will see when they click the sign-up button. The user will need to enter his or her full name, email address and password to create a new account. After that the users need to verify the CAPTCHA to prove that they are humans not bots. Last, they click the register button and the system will send them a confirmation email. The account will be created if they verify by clicking the link in the confirmation email.

5.6.2 Sign In

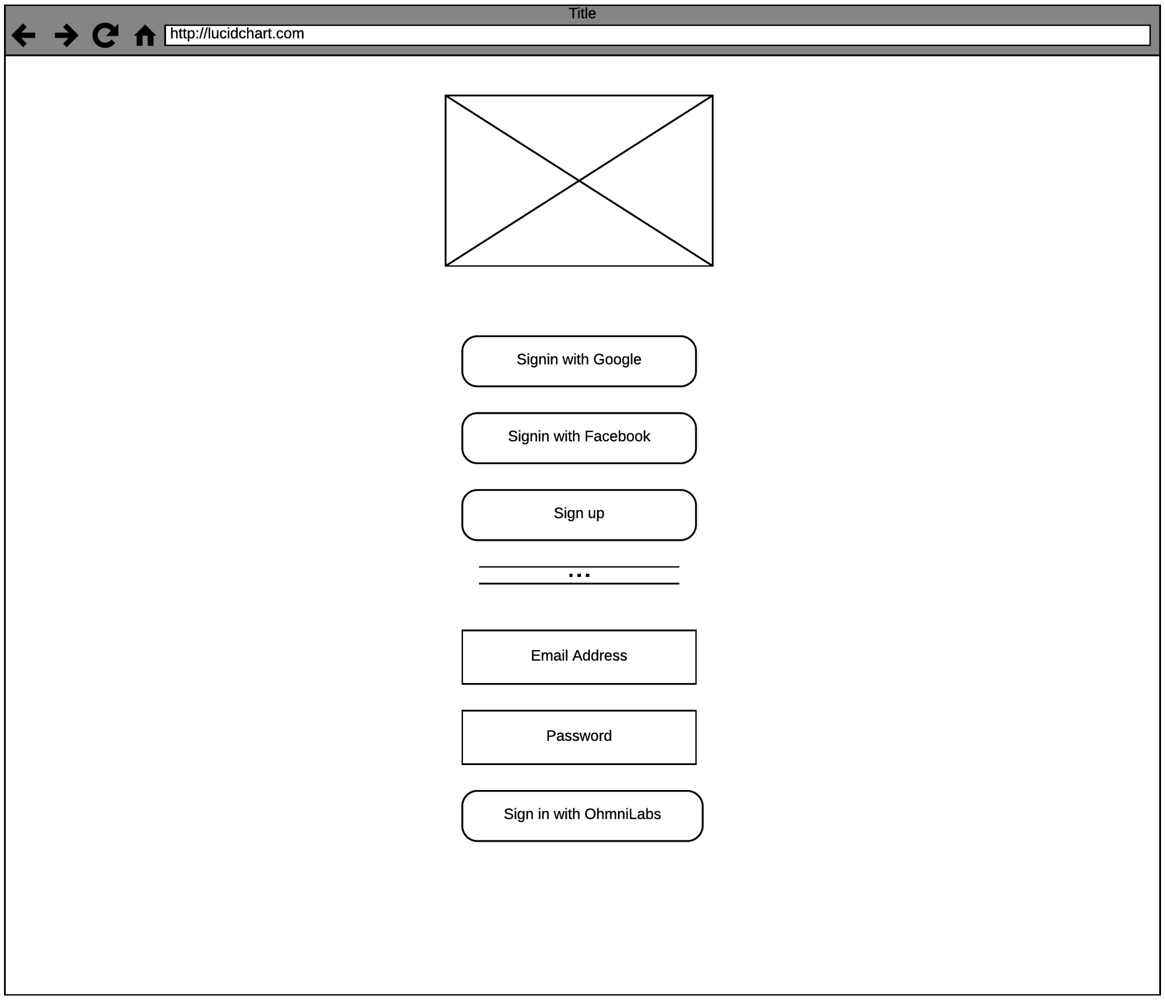


Figure 5 Sign in Mockup

This is the screen OhmniLabs displays to the users when they go to the main page. Users can sign-up with their OhmniLabs account by entering their email address and password. They can choose to login with a third party authentication like Google+ or Facebook. If they forget their password, they can click the forgot password button to reset their password.

5.6.3 Forgot Password

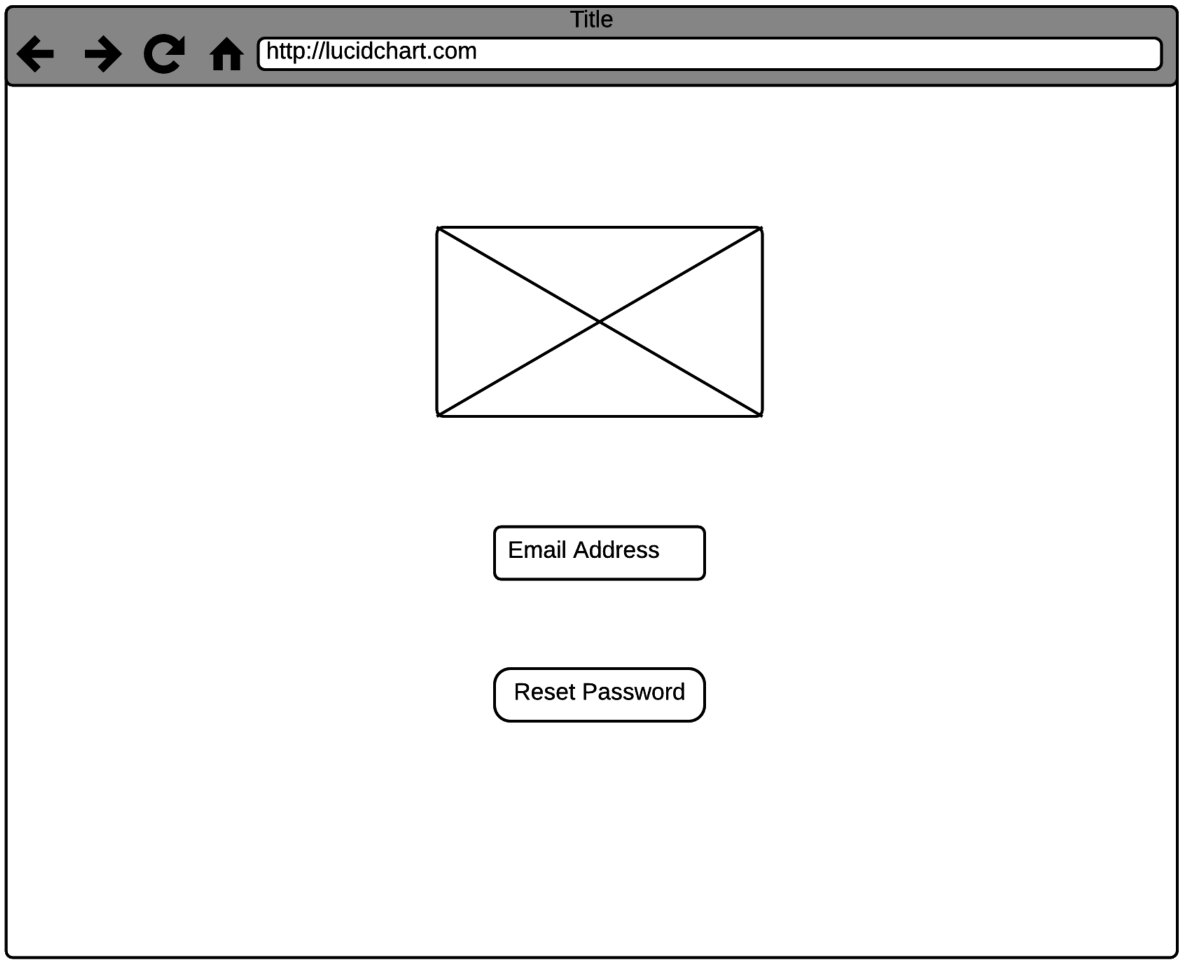


Figure 6 Forgot Password Mockup

This is the screen that appears to the users when they press the forgot password button on the main page. The user can enter their email address here and the system will send a link to their email address to reset their password if that email exists in the database.

5.6.4 Reset Password

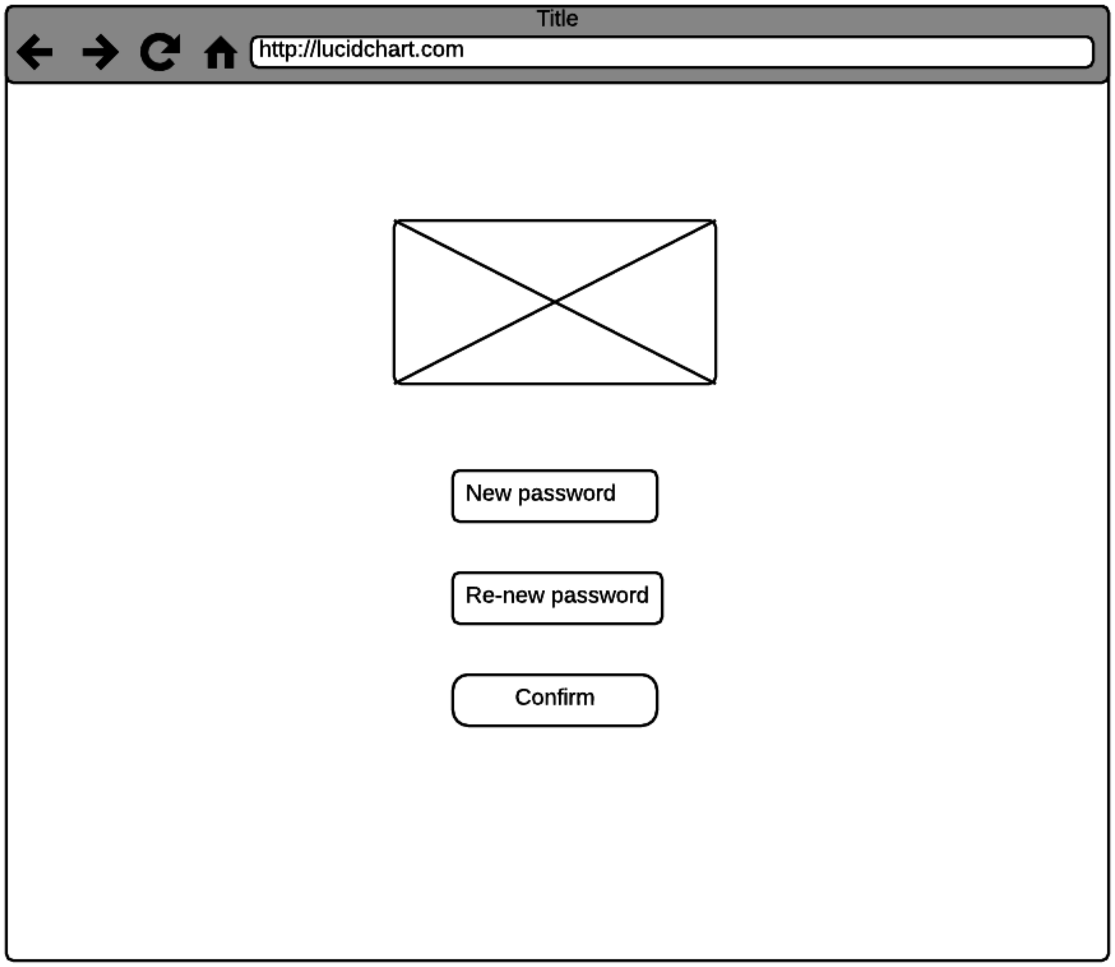
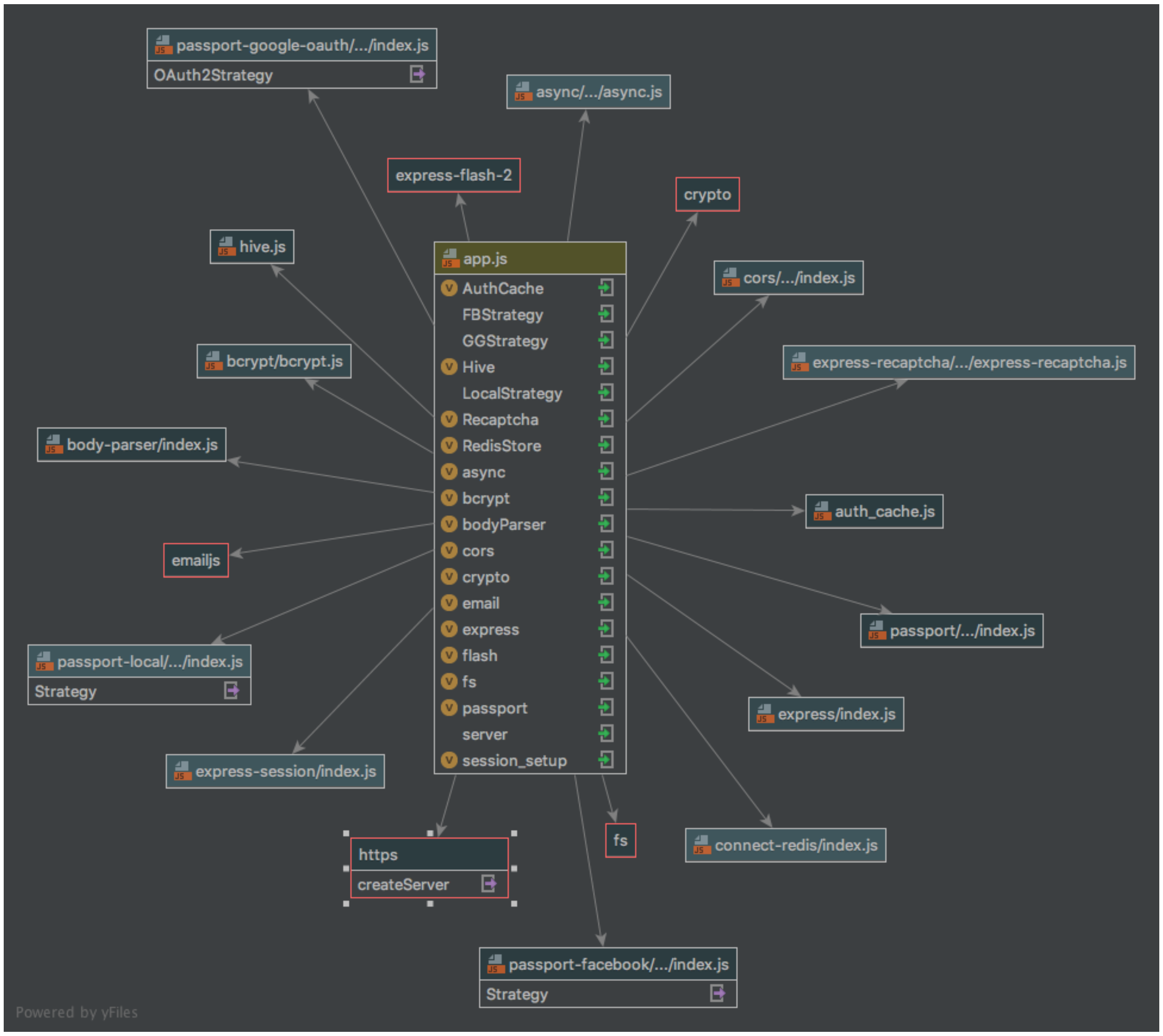


Figure 7 Reset Password Mockup

The users enter their new password and confirm it again. After both of the passwords match, the system sets it to become the user's new password.

6.0 Preliminary Design

6.1 Class Diagram



App.js

App.js is main configuration file for the Express app that contains every component of the OhmniLabs website. The file imports an object from another file or module. These objects have various functions such as ReCAPTCHA for the CAPTCHA in sign-up page, RedisStore for communication with the Redis database, flash for showing flash messages without re-rendering the website, crypto for encrypting passwords.

Express-reCAPTCHA

Express-reCAPTCHA is a Google reCAPTCHA middleware for Node.js. It gets CAPTCHA data from a query and verifies it with Google reCAPTCHA.

Passport.js

Passport is an authentication middleware for Node.js. It is flexible and can be combined into any Express-based web application.

Emailjs

Emailsjs is a middleware for Node.js that can send emails, html and attachments to any smtp server. It is used to send confirmation emails to users.

Express-flash

Express-flash is a middleware for Node.js that can define a flash message and render it without redirecting the request.

7.0 Implementation

This section outlines each iteration completed throughout the course of this project. Each iteration lasted from one to two weeks since the author were using Kanban methodology which is not strictly time boxed. Code testing throughout each iteration was primarily done manually and reviewed by OhmniLabs.

7.1 OhmniLabs Login Website

The first iteration goal was to intergrate the authentication module, Passport.js, with the current website. In the second iteration, a sign-up page was added to the website. The email confirmation tool was implemented in the third iteration. The fourth iteration added a password reset page to the website. Below is the description for each iteration of the OhmniLabs Login Website.

7.1.1 Iteration One

The goal of the first iteration was to integrate the current login page with Passport.js (an authentication middleware for Node.js). At the end of this iteration, a login page with local authentication, facebook-authentication and google-authentication using Passport was integrated to the existed website.

The first week was spent to get familiar with the existing OhmniLabs codebase and Jade. After that, the existing authentication was replaced with the new one, Passport.js. The first iteration was a successful start for the project.

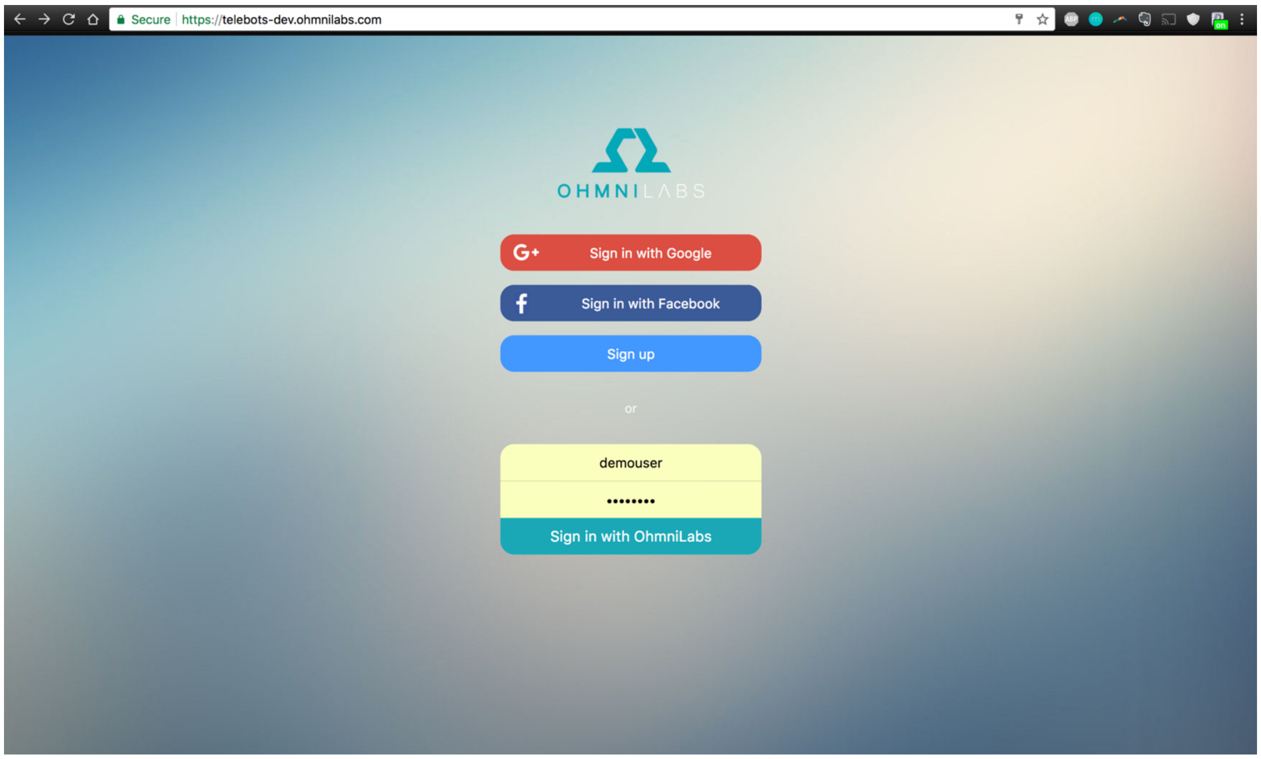


Figure 8 Login Page - Iteration One

7.1.2 Iteration Two

The goal of the second iteration was to implement the sign-up page. The existing design from the login page to implement a single sign-up page for OhmniLabs.

The majority of the development time was spent to read the codebase of the login page. By reusing the design of the login page, the sign-up page had a consistency design. After successfully implementing the sign-up page, the functionality was tested manually. A new user was created and tested in the database. At the end of the second iteration, a working sign-up page was successfully implemented.

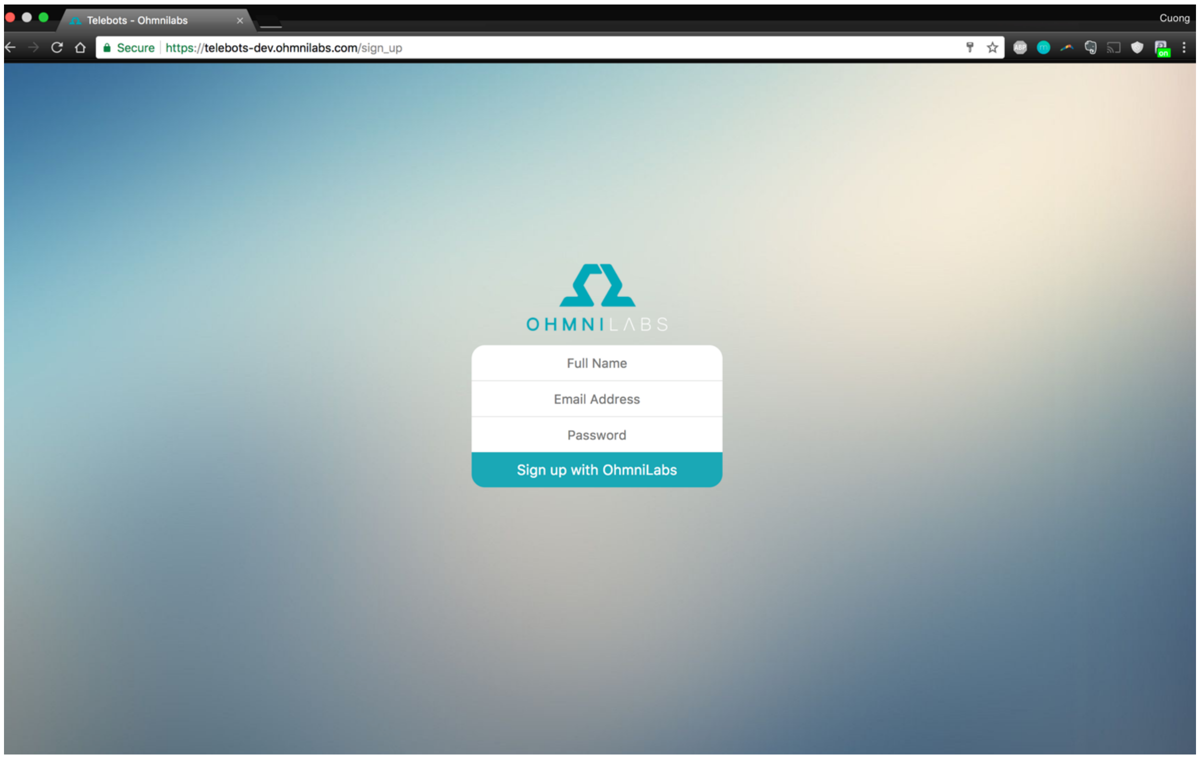


Figure 9 Sign-up Page

7.1.3 Iteration Three

The goal of the third iteration was to implement the email confirmation tool for the account creation and add reCAPTCHA for the sign-up page.

The reCAPTCHA was added to prevent bots spamming account. It was implemented by using the API from Google reCAPTCHA. However, there was a bug that the server did not get notifed even if the user has check the reCAPTCHA box.

The email confirmation tool was implemented by using a third party node package manager (npm) called emailjs. When the user clicked the sign-up button, it will send an email to the user email address and set the user account to lock state until the user click the link in the verified email.

At the end of the third iteration, the email confirmation tool was ready to use. However, the bug in the reCAPTCHA part was unresolved and carried on to the next iteration.

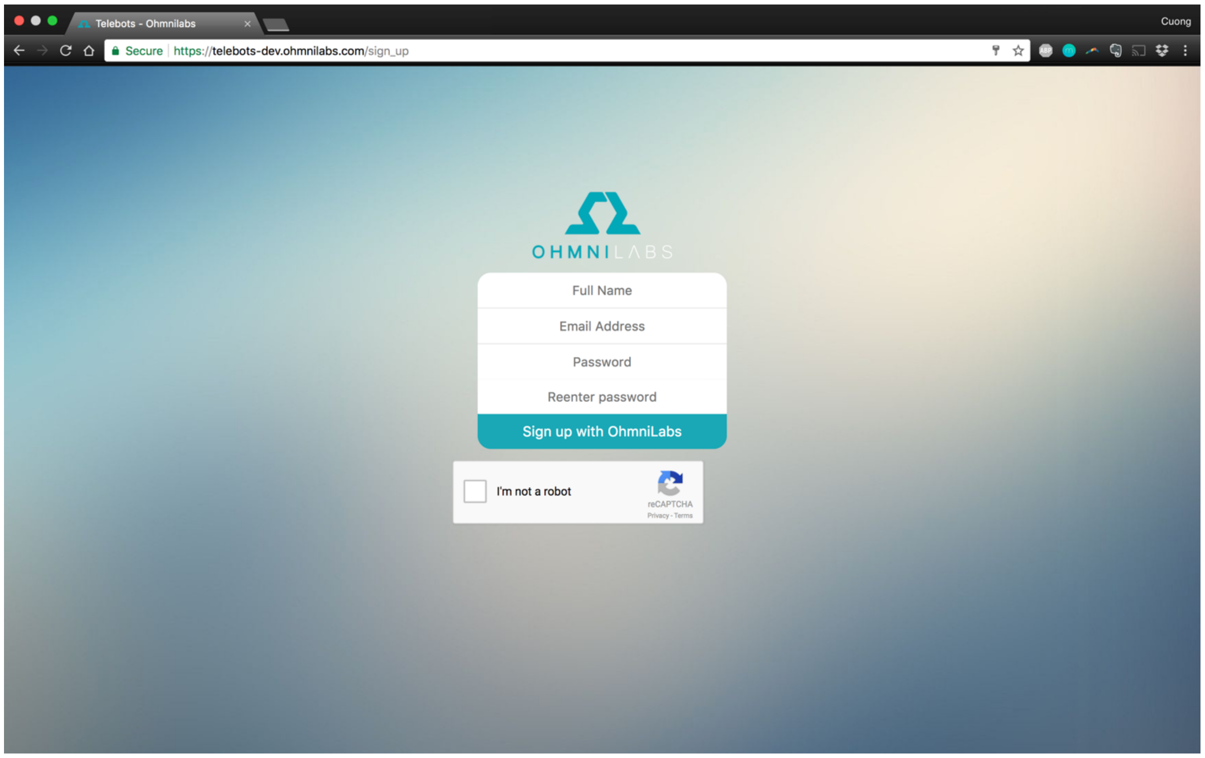


Figure 10 Sign-up Page with reCAPTCHA

7.1.4 Iteration Four

The goal of the fourth iteration was to add the password reset page to the login website and fix the reCAPTCHA bug.

Using the email confirmation tool from the last iteration, the password reset page sends a link to the user to redirect them to the password reset page. As a result, the user can set a new password for their account.

When a user clicks the Forgot Password button, the website redirects them to a single page that contains a text box for the user to enter their email address. The server encrypts the email address and adds a random hash code to the account. After that, it generates a link and sends an URL in an email message to the user.

At the end of this iteration, a user is able to reset his password in case it is forgotten. After testing multiple times, the reset password page was fully functional at the end of the iteration.

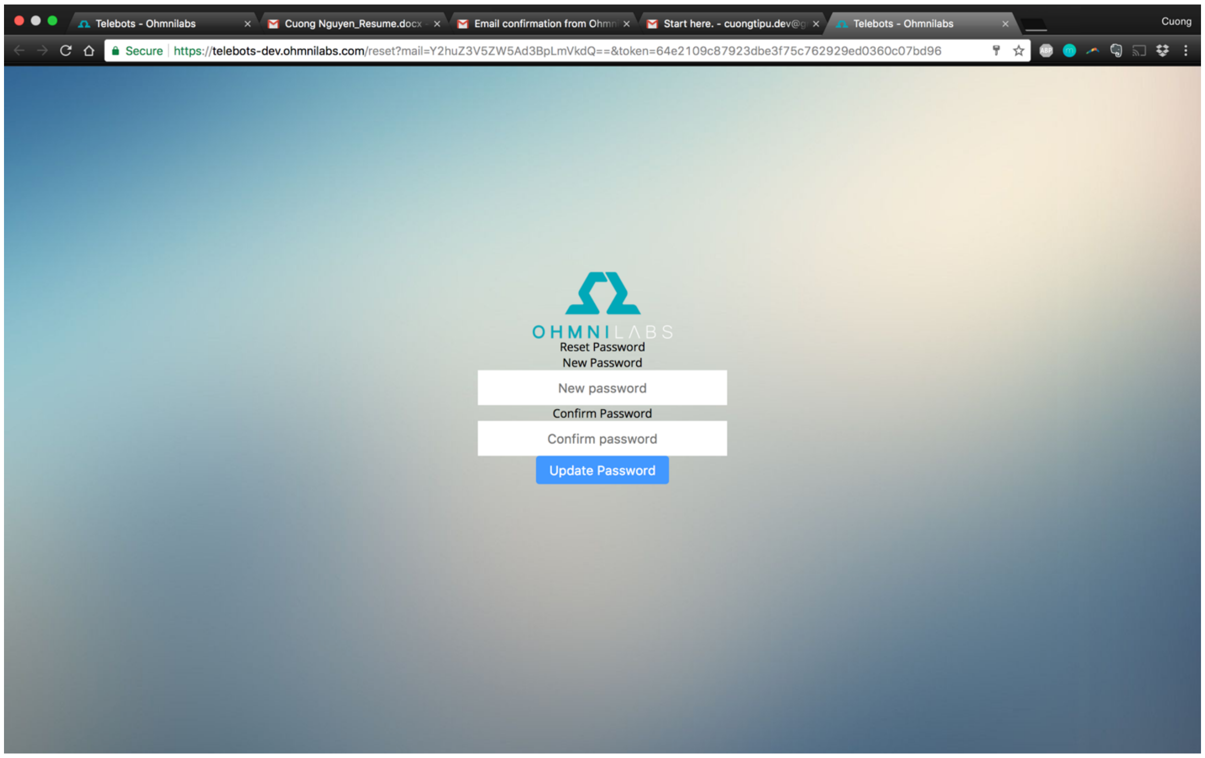

Telebots - Ohmnilabs
Cuong Nguyen_Resume.docx
Email confirmation from Ohmn
Start here. - cuongtipu.dev@g
Telebots - Ohmnilabs
Cuong
Secure
https://telebots-dev.ohmnilabs.com/reset?mail=Y2huZ3V5ZW5Ad3BpLmVkdQ==&token=64e2109c87923dbe3f75c762929ed0360c07bd96
OHMNILABS
Reset Password
New Password
New password
Confirm Password
Confirm password
Update Password

Figure 11 Reset Password Page

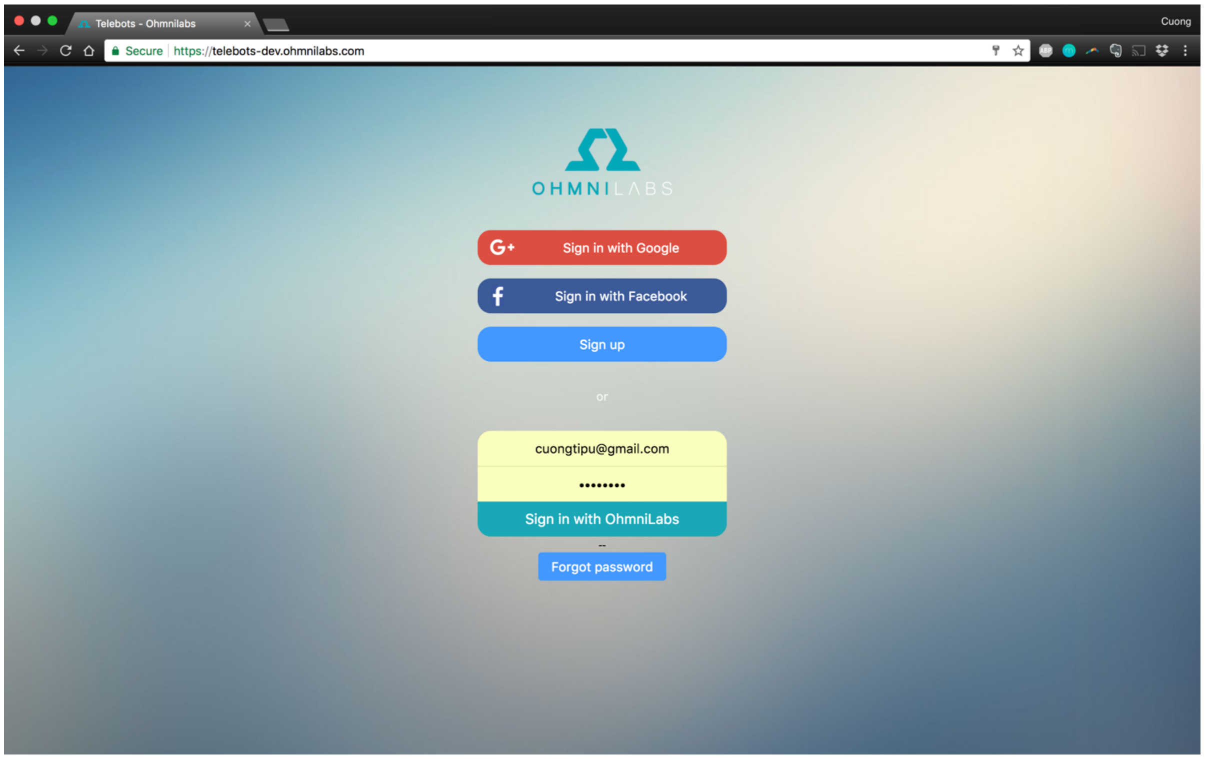


Figure 12 Login Page - Iteration Four

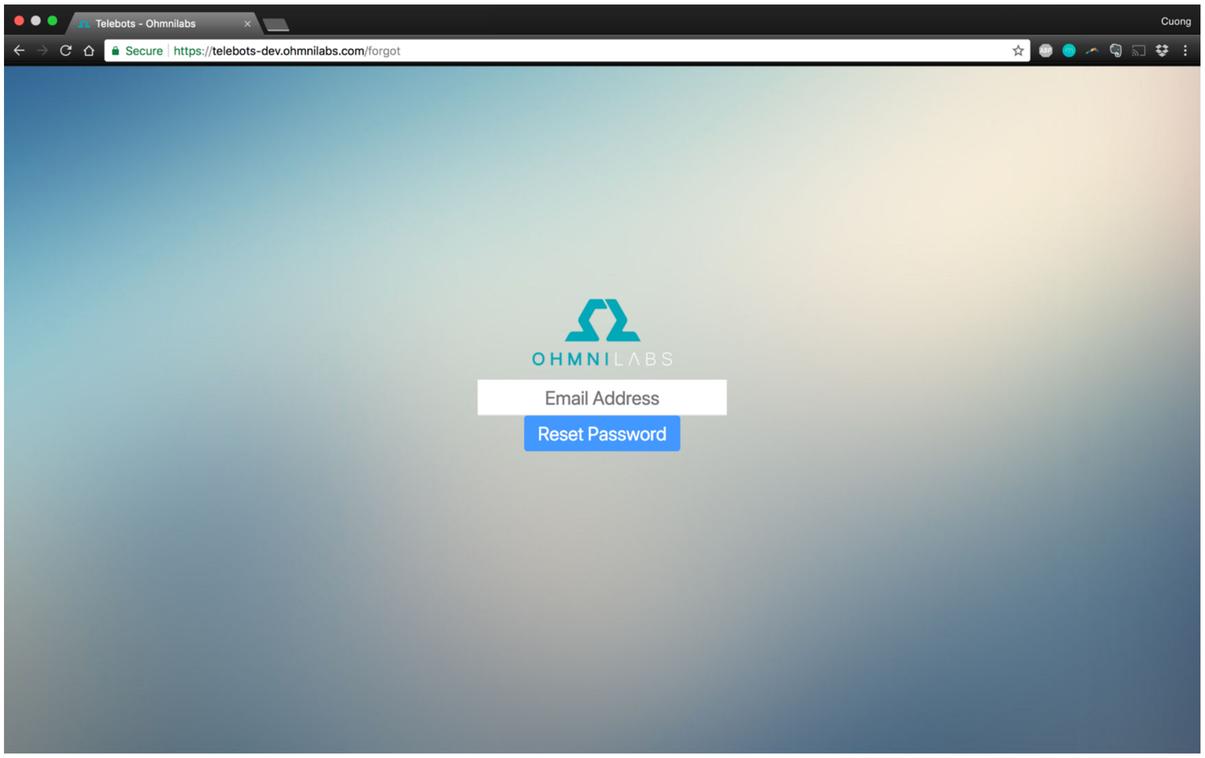


Figure 13 Email Address to Reset Password

8.0 Assessment

This section looks back on the project as a whole and reflects upon the project's progress from start to finish.

8.1 Accomplishment

The project goal was to build an authentication module for OhmniLabs video call website. At the end of the project, the functionality of account creation, account retrieval and account authorization was successfully implemented. Google reCAPTCHA was used to protect the website from spam bots. An account email confirmation tool was created to validate the user email address. Both of the reCAPTCHA and email confirmation tools helped to avoid spamming and increase security for the website.

8.2 Lessons Learned

The education value from developing this project is unparalleled to any other projects completed at school. Full web development requires knowledge of many different technologies such as back-end development, front-end development, user experience, documentation and database. Each of the components has different programming languages and frameworks that needs to be understood to deliver a functional project.

The project provides a real life working experience in a startup environment. Lessons about coding style, code reviews and code design were a valuable learning experience for the author. Working individually also provided the author with the challenge of time management for the project.

8.3 Areas Needing Improvement

Testing should be more formalized and thorough. Research into testing for Javascript needed to be done to improve the quality of the project. The majority of the testing was done manually since much of the project involved user interface design. Unit testing should be made for further features in the future.

Working as a single developer on a project of this scale is difficult. There was a great deal for one person to balance all of the components of the project. In hindsight, the author needed to be more experienced in web development to handle every task of the project. Although the author struggled when developing and learning simultaneously, most of the functional requirements were successfully implemented.

9.0 Future work

This section will outline features that were planned to be implemented, but due to time constraints were not completed, as well as other functionality suggested to improve the application.

iOS supporting

Supporting for iOS devices would be a major feature for this project. At this moment, user can only video call through an Android device. Having iOS browser support could attract more users to use the company products.

Mobile application

Mobile compatibility could be another solution to improve user experiences. An iOS application or Android application might bring a better way for users to use the video call function. The project might be created with a separate version using React Native. A great benefit that React Native brings is the application would be deployable on both Android and iOS.

10.0 Conclusion

The goal of the project was to implement account authorization, account creation and account retrieval for the OhmniLabs website. Although not every goal of the project was met during the time constraint, the major functionality was successfully implemented. Users can create a new account with their email address. The OhmniLabs website has a mechanism for the user to reset their password in case the user forgets their password. An email confirmation tool was implemented and can be used for various purposes when the website wants to validate the email address.

Overall, the OhmniLabs project met most of its critical goals although some of the features such as admin dashboard, sharing and invitation could be added in the future. This was an extremely invaluable experience on web technologies and startup working environment for the author.

Bibliography

Atlassian. (2016). The agile coach. Retrieved from https://www.atlassian.com/agile/kanban

Choudhry, A. (2016). Is WebStorm the Smartest JavaScript IDE around? Retrieved from https://medium.com/@AliyahChoudhry/is-webstorm-the-smartest-javascript-ide-around-8a5831e5842f

Duffy, J. (2016). Trello. Retrieved from https://www.pcmag.com/article2/0,2817,2487001,00.asp

Ilic, J. (2014). The History of Vim. Retrieved from https://jovicailic.org/2014/06/the-history-of-vim/

Justin. (2013). What is the software development life cycle (SDLC)? Retrieved from https://airbrake.io/blog/sdlc/what-is-the-software-development-life-cycle

LaGreca, K. (2018). How We Use It: Slack As A Team Communication Tool. Retrieved from https://www.npgroup.net/blog/how-we-use-it-slack-as-a-team-communication-tool/

LeanKit. A physical Kanban board with a basic, three-step workflow. Retrieved from https://leankit.com/learn/kanban/kanban-board/

Putnam-Majarian, T., & Putnam, D. (2015). The Most Common Reasons Why Software Projects Fail. Retrieved from https://www.infoq.com/articles/software-failure-reasons

Shelly, & Cashman. (2009). *Systems analysis and design* (8th ed.). Boston, MA: Thomson Course Technology.

Shon, P. (2014). Redis Explained in 5 minutes or less. Retrieved from https://www.credera.com/blog/technology-insights/java/redis-explained-5-minutes-less/

Tan, T. (2017). OhmniLabs Introduces Ohmni, A Home Robot that Transforms How Families Stay Connected. Retrieved from https://blog.ohmnilabs.com/ohmnilabs-introduces-ohmni-a-home-robot-that-transforms-how-families-stay-connected

Techopedia. (2017a). Collaboration Software. Retrieved from https://www.techopedia.com/definition/6542/collaboration-software

Techopedia. (2017b). Project Management Software. Retrieved from https://www.techopedia.com/definition/13132/project-management-software

TutorialsPoint. (2016). SDLC-overview. Retrieved from https://www.tutorialspoint.com/sdlc/sdlc_overview.htm

What is agile? What is Scrum? (2013). Retrieved from https://www.cprime.com/resources/what-is-agile-whatis-scrum/