ResearchMate

Software Design

CSCI-P465/565 (Software Engineering I)

Project Team

Gulshan Madhwani

Shubham Basu

Jayendra Khandare

Xinquan Wu

---

1. Introduction

This document is a high-level description of the system that we are building – ResearchMate. ResearchMate is a web application that let PhD researchers connect to other PhD students and advisors. This document provides explanation of the system design, design approach, system architecture and component design of ResearchMate.

1.1 System Description

ResearchMate is a web application, a collaborative platform where researchers can connect with other researchers that share a common interest. It is an easy to use, user friendly way of posting doubts, conducting a group discussion, having live chat. It also provides a bullet-in board where user can get latest updates, trends in their area of interest. This application targets the users who find it difficult to connect with people of his/her interest. Having a group discussion with people having different perspective can help them conduct their research smoothly and efficiently by considering all possible scenarios.

1.2 Design Evolution

1.2.1 Design Issues

Due to lack of experience in MEAN stack, we thought of proceeding with .NET but we don't have place to host .NET application on silo machine. So we stick to MEAN stack and started learning.
Issues we faced in the past two weeks:

- Whether to program front end using AngularJS or jQuery. We decided to go with AngularJS since it is good for single page application development.
- We are also trying to make the application responsive using Bootstrap. Considering Bootstrap is new to most of the members, it is taking time to make a good and robust design.
- MEAN stack being a new technologies to learn, we were not comfortable with making service requests from AngularJS to NodeJS.
- NodeJS provides lots of packages to use that eases the development process. Being new to NodeJS, we had a hard time searching for an appropriate module. For example, nodemailer for sending emails is most widely used but is not very secure.
- For Duo authentication, we are generating a random key which is sent to the user through email (as a link). Once user clicks the link, we authenticate the user on backend.

1.2.2 Candidate Design Solutions

With some research, MEAN stack came out to be a perfect stack of technologies for our project. Our application mainly consists of CRUD operations. MEAN provides all the required technologies in one stack – MongoDB for database, AngularJS for front end, NodeJS for backend, ExpressJS facilitates communication between AngularJS and NodeJS.

- We are currently exploring more appropriate and efficient NodeJS modules for our application useful in various parts of the application. This can help us in future and design a system that is scalable and easily manageable.
- We are also exploring how we can perform Duo authentication using Google Duo, similar to CAS login. Currently we are performing Duo authentication using random key generation and authenticating using email address. We are currently exploring a NodeJS module called "passport" for this task.

1.2.3 Design Solution Rationale

- We are currently using ‘nodemailer’ module in NodeJS for emailing. It is the most widely used and has good amount of online documentation. Though not very secure as compared to other modules, using this module reduces a good amount of time of development and serves an initial purpose of implementing Duo authentication.
- Implementing Duo authentication using Google services is time consuming and requires some initial monetary investment as well. We are currently using authentication through email which serves the purpose of verifying if the user is authentic or not. In future, if time permits, Google Duo can be implemented.

1.3 Design Approach

1.3.1 Methods

The primary feature of MEAN stack is abstraction that it provides. Each technologies provides a layer of abstraction. Service calls are made from AngularJS using Express which in turn makes the actual call to NodeJS. NodeJS provides interfaces to connect to the database and performs actual CRUD operations. Each layer works independently which allows different members work independently.
We are also using various node modules that makes the application efficient, secure, and scalable.

1.3.2 Standards

- We are using Bootstrap for UI development since it is efficient, user friendly, and responsive.
- We are using AngularJS for front end development because it’s good for single page application development.

1.3.3 Tools

Front end:

- Programming: AngularJS

- IDE: Atom/Visual code

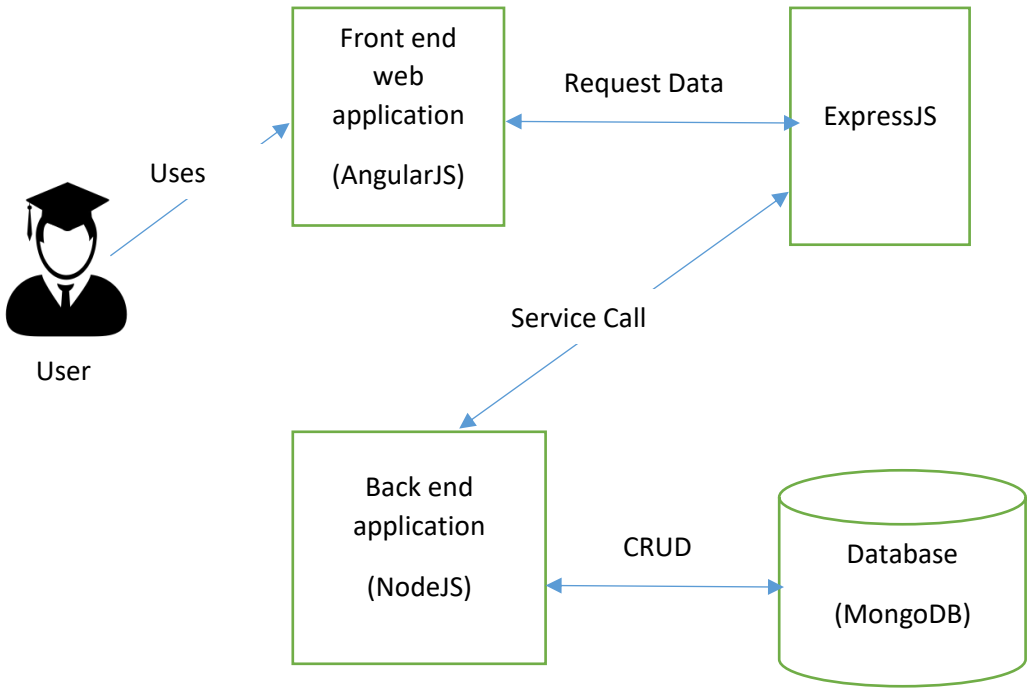
Back end:

- Programming: NodeJS
- IDE: Atom

Build: Build using webpack. It will allow to create a single JS file for production and eliminates the need to managing many JS files.
Database: MongoDB
Hosting: silo.indiana.edu
Source Control: GitHub
Coordination: JIRA

2. System Architecture

2.1 System Design



User: User is any person having a profile on our web application. User can be a researcher, advisor, admit, etc.

Front end: Front end application developed using AngularJS, Bootstrap, HTML, and CSS provides a basic user interface where user can login, sign up, view bullet-in, participate in discussion, etc.
Back end: Backend is developed using NodeJS. This component provides an interface between database and front end and facilitates to perform CRUD operations on the database.
Database: This is where all the data is stored. We are using MongoDB for database.

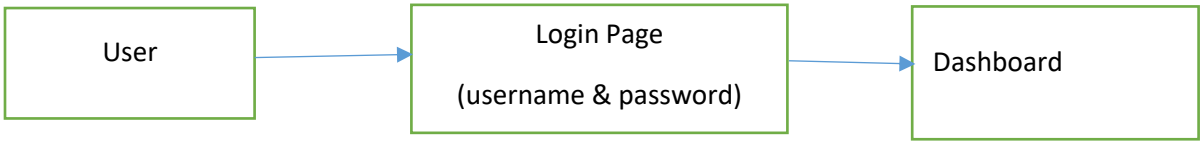
2.2 External Interfaces

External interfaces for our application are users. These users are classified as PhD students/researchers, advisors. Researchers can post on discussions page, join group, view updates on bullet-in board, and share their profile. Advisors can create groups for students to join and can approve their requests.

3. Component Design

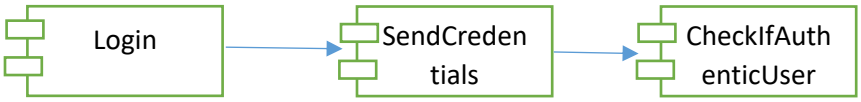
Component 1:

- **Component Name**
  Login Component
- **Component Description**
  This component allows the user to login by providing username and password.
- **Responsible Development Team Member**
  Shubham, Jayendra, Gulshan.
- **Component Diagram**



- **Component User Interface**
  For login, user is supposed to provide username and password.

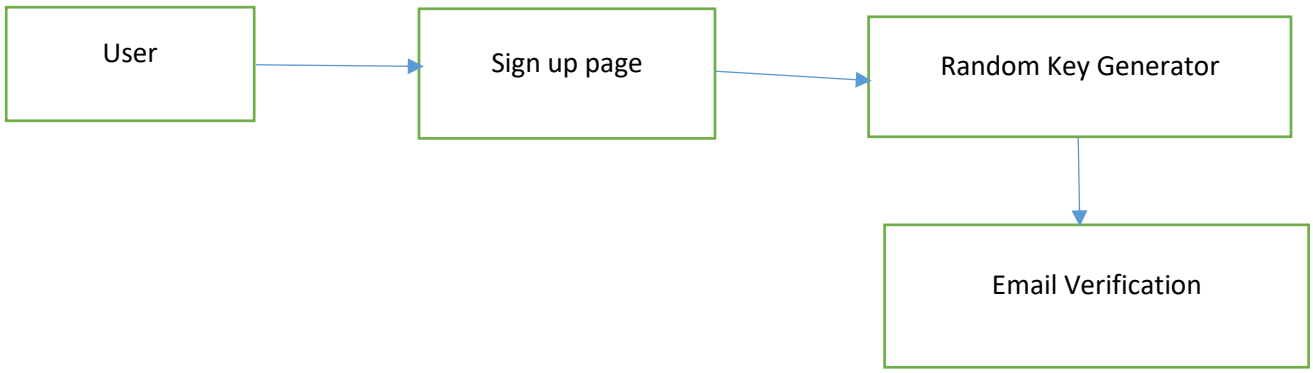
- **Component Objects**
  classes: Login, SendCredentials, CheckIfAuthenticUser



- **Component Interfaces (internal and external)**
  For login, user provides username and password using the text boxes provided on the login page. These credentials are then passed to the server side where the user is authenticated and checked if the profile already exist. If profile exists, the user is redirected to Dashboard else the user can sign up and create a profile.
- **Component Error Handling**
  Username and password is checked for blanks and valid email address.

Component 2:

- **Component Name**
  Sign up
- **Component Description**
  This component allows the user to sign up. User needs to provide email address, first name, last name, and password. User is authenticated using random key generation and email verification.
- **Responsible Development Team Member**
  Jayendra, Xinquan.
- **Component Diagram**



- **Component User Interface**
  For sign up, user needs to provide first name, last name, email address, and password using the text boxes provided on the sign up page.
- **Component Objects**
  GetUserInput, CheckValidInput, GenerateRandomKey, EmailKey, VerifyUser
- **Component Interfaces (internal and external)**
  GetUserInput takes the input from user form.

  CheckValidInput checks if the entered input is valid and all the required fields are provided.

  GenerateRandomKey generates a random key for email authentication.

  EmailKey is a node module that email the randomly generated key to the user.

  VerifyUser is a module that checks if the key provided for verification is same as the key generated for that user.

- **Component Error Handling**
  Check for all the required fields: First name, last name, email address and password.
- Email address verification: checking the format and authenticating it using random key generation.

Revision History

Revision	Date	Change Description
Sprint 1	10/01/2017	Module: Login & Sign up using Duo authentication.

Page Author/Creator: Adeel Bhutta
Last Modified: 8/23/2016