Needless Youth Unemployment System Requirements Specification (SRS)

Version 1.2

Document Number: SRS-001

Project Team Number: A02
Project Team Members:

Leo Liu	yl5898
Rihui Zheng	rz1276
Md Abedin	mka374
Kyle Lin	kl3399

REVIEW AND APPROVALS

Team Members	**Function**	**Date**	**Signature**
Leo Liu	Co-author	November 19, 2020	Leo Liu
Md Abedin	Co-author	November 19, 2020	[illegible]
Rihui Zheng	Co-author	November 19, 2020	Rihui Zheng
Kyle Lin	Co-author	November 19, 2020	Kyle Lin

REVISION LEVEL

Date	Revision Number	Purpose
10/6/2020	Version 1.0	Initial Release
10/21/2020	Version 1.1	Define Project Requirements
11/19/2020	Version 1.2	Complete Document

Table of Contents

Table of Contents **4**

1. Document Purpose **6**
1.1 Purpose 6

2. Introduction **7**
2.1 Scope 7
2.2 Identification 7
2.3 Bounds 7
2.4 Objectives 7
2.5 Context Diagram 8
2.6 Additional Descriptive Items 8

3. Glossary **10**

4. Reference Documents **11**

5. Business Requirements **12**
5.1 Technology 12
5.2 Economics 12
5.3 Regulatory and Legal 12
5.4 Market Considerations 12
5.5 Risks and Alternatives 12
5.6 Human Resources and Training 12

6. User Requirements **13**
6.1 Functional Descriptive Detailed Requirements 13
6.2 Non-functional Descriptive Detailed Requirements 13

7. System Architecture **14**

8. Detailed System Requirements — Use Cases **15**
8.1 Requirements Use Cases 15
8.1.1 Use Case DiagramsAu 15
8.1.2 Use Case Descriptions 15

9. System Model (UML) **18**
9.1 Static - Class Diagrams 18
9.2 Dynamic - Behavioral Models 20

10. Evolution of the SRS **27**

11. Rationale **28**

12. Notes **29**

13. Appendices **30**
13.1 System Test Plan Requirements 30
13.2 Qualification Provisions 30
13.3 Requirements Traceability 31
13.4 Schedule Tracking 31
13.5 Defect Tracking 32
13.6 Dictionaries 33
14. Index **39**

1. Document Purpose

1.1 Purpose

The purpose of this document is to specify the requirements of the Needless Youth Unemployment software project. The intended audience of this document is the clients, creators, and stakeholders involved with the project.

2. Introduction

2.1 Scope

- The system shall scrape websites of companies to notify the user of job or internship opportunities.
- The system shall allow the user to upload a resume and add their interests.
- The system shall show the user the statistics and the results of their applications.
- The system shall be integrated with a messaging application as a bot to notify the user.
- The system shall autofill fields for users when they fill out the application

2.2 Identification

This is version 1.1 of the System Requirements Specification document.

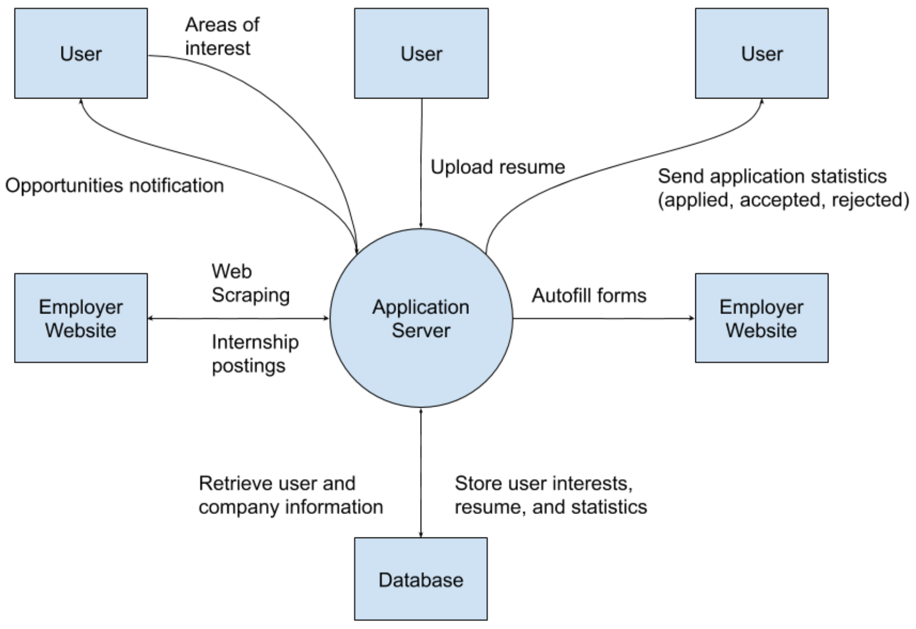
2.3 Bounds

The system will not be compatible with all companies which post job or internship opportunities. Only companies that are desirable will be scraped with the system.

2.4 Objectives

This will be a high priority project that will be delivered as a single deliverable on December 14th, 2020.

2.5 Context Diagram


User
Areas of interest
User
User
Opportunities notification
Upload resume
Send application statistics (applied, accepted, rejected)
Employer Website
Web Scraping
Internship postings
Application Server
Autofill forms
Employer Website
Retrieve user and company information
Store user interests, resume, and statistics
Database

2.6 Additional Descriptive Items

a. Product functions
    - Scrapes information on internships and jobs from company websites
    - Shows user available internship/job applications
    - Allows the user to upload resumes, their interests, and other information necessary for their application
    - Autofills user information on internship/job applications
    - Shows users their application results and statistics
    - Pings the user on the messaging application if certain companies of interest open their application for internships or jobs

b. User Characteristics
    - Anyone who is looking for internships or jobs

c. Constraints
    - Certain company websites may have policies against scraping their website for information
    - Having the resources to keep the program running at all times
    - Since users are sending their information to our server so our server can autofill applications, we need to have good security to protect user information

d. Assumptions and dependencies
    - Companies' career websites are scrapable for application information

- Users are willing to host their resume and private information on our server
- Constant internet connection is required

e. Requirements subsets
- Not applicable

3. Glossary

Term	**Definition**
API	Application Programming Interface. Defines standardized interactions between software.
Scrape	Automated data extraction from an interface meant for human use to feed into another program.
Ping	The term used to describe a notification in the messaging application.
Bot	A specific type of user in the messaging application signifying that there is no person behind it.

4. Reference Documents

Team A02, Project Proposal, Version 1, Sept 21, 2020

5. Business Requirements

5.1 Technology

Our product shall be capable of scraping the various company websites and utilize the messaging application’s bot.

5.2 Economics

Our product shall assist people in finding job and internship opportunities with more ease, which will indirectly affect the economy.

5.3 Regulatory and Legal

Our product shall follow the terms and conditions of each prospective business websites and media outlets that we web scrape.

5.4 Market Considerations

Our product shall be used by anyone who is looking for an internship or a job.

5.5 Risks and Alternatives

A major risk for our product is there could be some companies that have terms of use that prohibit the scraping of their website. There is no programmatic alternative to this feature, so the user will either have to check the website themselves or not check that website at all. Our product could let the user know if any of the companies we recommend for them have these policies and encourage the user to check out the company's website themselves.

5.6 Human Resources and Training

Not applicable.

6. User Requirements

6.1 Functional Descriptive Detailed Requirements

User Requirements:

- A user shall be able to mark areas and companies of interest.
- A user shall be notified of internship and job opportunities on a messaging application as they open.
- A user shall be able to autofill job applications.

System Requirements:

- The system shall scrape websites each day, for each company, and generate a list of openings.
- The system shall track user accept, reject, and no-reply statistics.
- The system shall save the user's resume in the database.

6.2 Non-functional Descriptive Detailed Requirements

Product Requirements:

- The system shall be available to all clients at all times. Downtime shall not exceed 30 minutes in any one day.
- The system shall notify the user within 10 minutes of receiving an updated status.

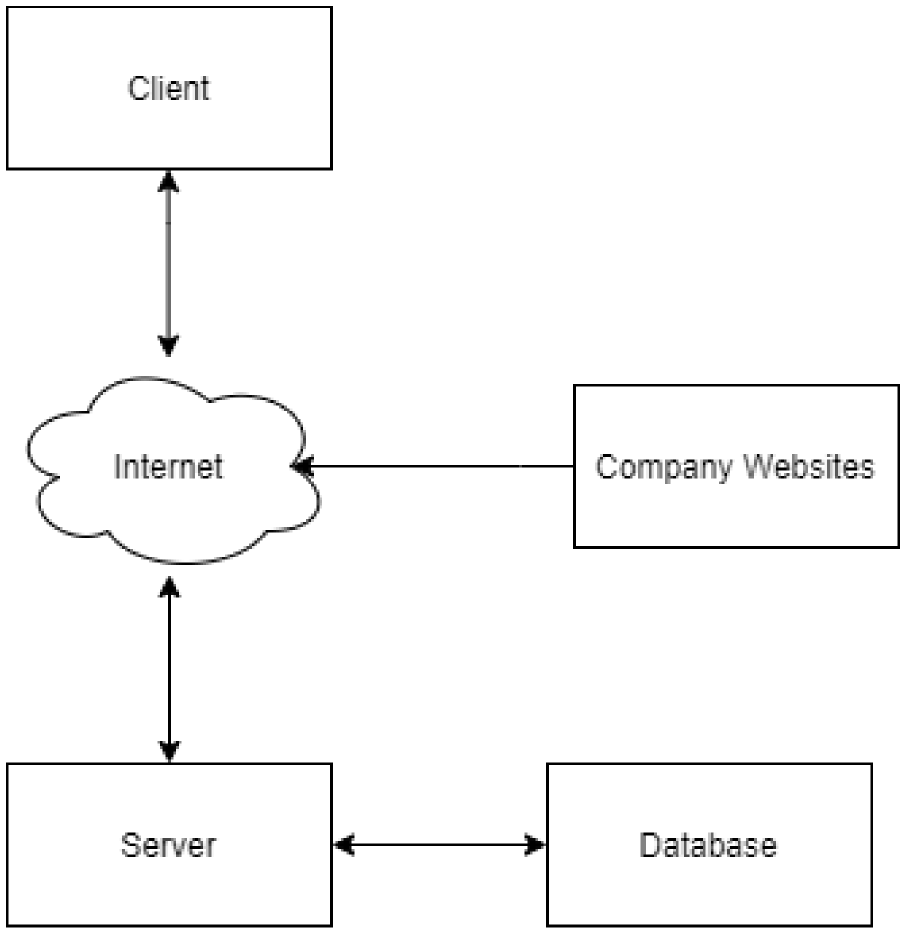
Organizational Requirements:

- Users of the system shall authenticate themselves using a passcode.

External Requirements:

- The system shall not share information about the user with any persons except the user.
- The system shall not store confidential information in plain text.

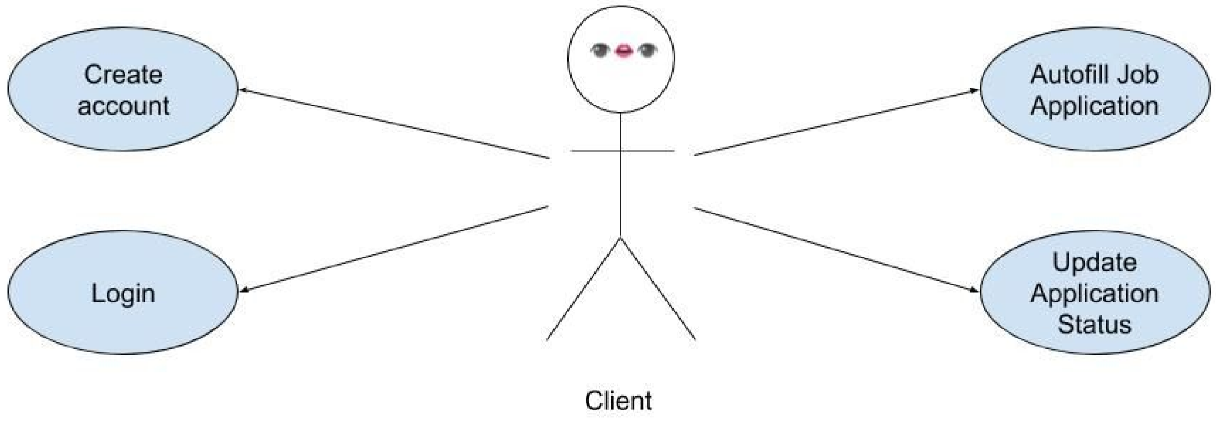
7. System Architecture



8. Detailed System Requirements — Use Cases

8.1 Requirements Use Cases

8.1.1 Use Case Diagrams



8.1.2 Use Case Descriptions

Create Account		
Description	A user can create an account through the client interface	
Pre-Conditions	The user is not already logged in	
Flows	Basic or Normal Flows	1. The clicks on the "Register" button 2. The user creates a username and a password 3. The user clicks on the "Create Account" button
	Alternative Flows	None
Post Conditions	The user has an associated account in the database and is logged in.	
Special Requirements	None	
Extension Points	None	

Login		
Description	A user can log in to their account	
Pre-Conditions	The user is not already logged in and has an account	
Flows	Basic or Normal Flows	1. The user clicks on the "Login" button 2. The user enters their username and password 3. The user clicks on the "Login" button
	Alternative Flows	After step 3, if the username and password combination is not found in the database, restart from step 2.
Post Conditions	The user is now logged in.	
Special Requirements	None	
Extension Points	None	

Autofill Job Application		
Description	The user will choose a job and the system will fill out fields of the job application form for them automatically	
Pre-Conditions	The user needs to have an account and some of their application info saved in the system	
Flows	Basic or Normal Flows	1. The user logs in 2. The user chooses a job and clicks the autofill option 3. The system loads their info into the fields it can recognize within the job application
	Alternative Flows	None
Post Conditions	The job application they selected will have some fields filled out	
Special Requirements	None	
Extension Points	None	

Update Application Statuses		
Description	The user reports the status of their job application for their personal statistical tracking	
Pre-Conditions	The user heard back from potential employers after applying to an opportunity	
Flows	Basic or Normal Flows	1. The user selects one of their active applications 2. The user reports its outcome 3. The outcome is stored within the database
	Alternative Flows	None
Post Conditions	The database now holds the status of the user's application.	
Special Requirements	None	
Extension Points	None	

9. System Model (UML)

9.1 Static - Class Diagrams

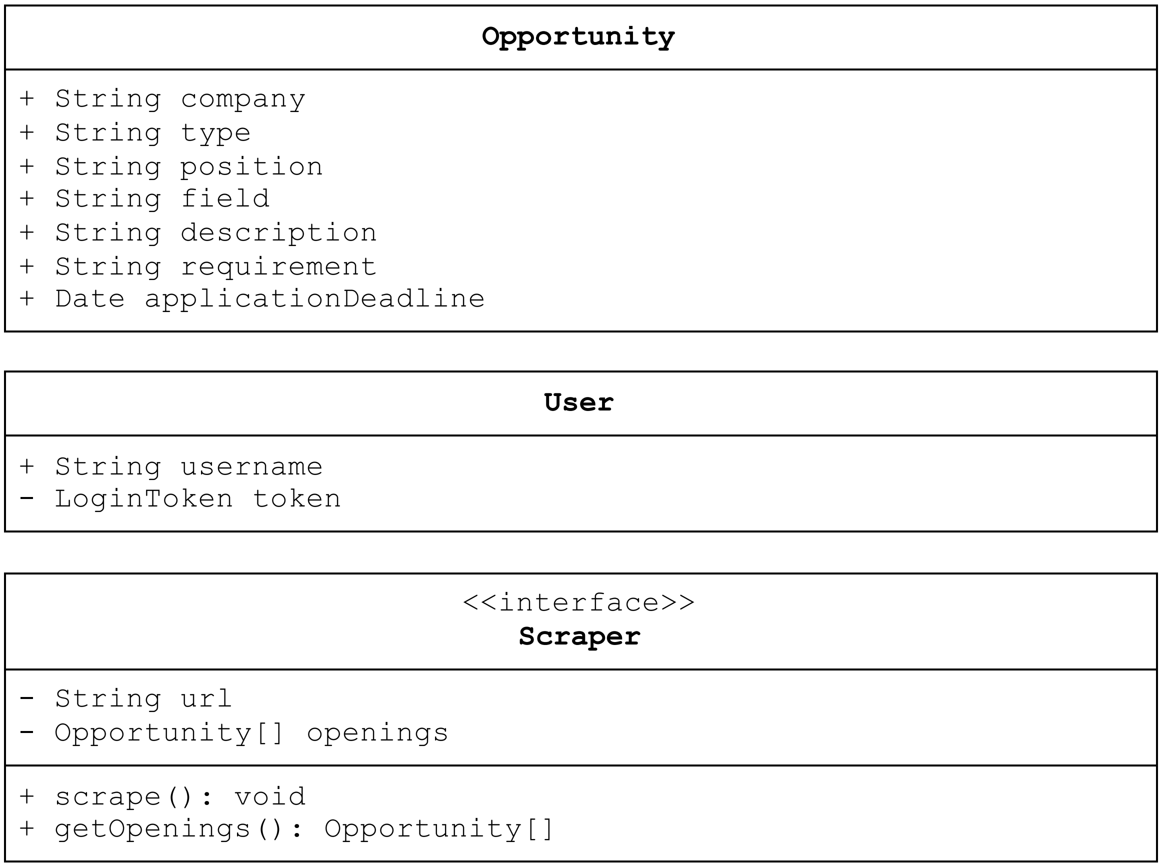
Interfaces:

- Scraper
- DatabaseManagementSystem

Candidate Classes:

- User
- Opportunity

Class Diagram:



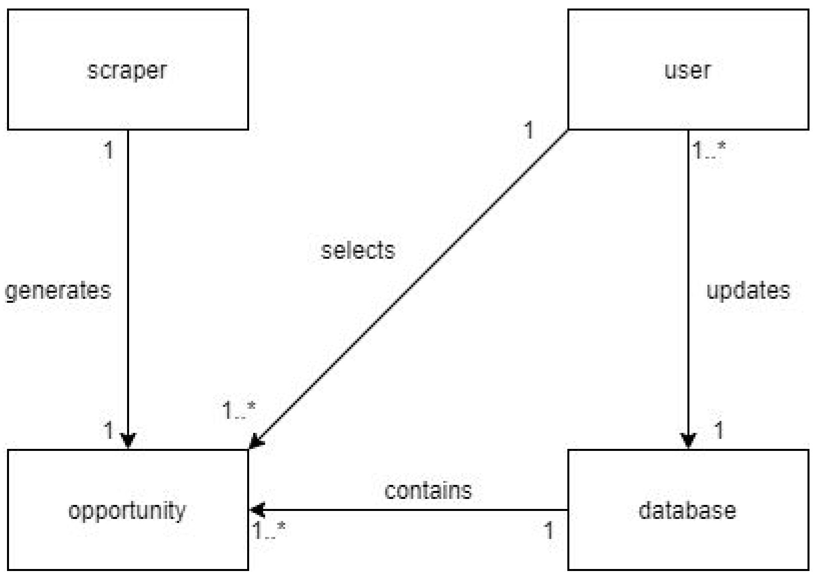
```
Opportunity
---
+ String company
+ String type
+ String position
+ String field
+ String description
+ String requirement
+ Date applicationDeadline
```

```
User
---
+ String username
- LoginToken token
```

```
<<interface>>
Scraper
---
- String url
- Opportunity[] openings
---
+ scrape(): void
+ getOpenings(): Opportunity[]
```

```
<<interface>>
DatabaseMangementSystem
---------------------------------------------------------------
- Connection conn
---------------------------------------------------------------
+ validateCredential(): User
+ createAccount(): User
+ getOpportunityByType(String type): Opportunity[]
+ getOpportunityByCompany(String company): Opportunity[]
+ getOpportunityByField(String field): Opportunity[]
+ getApplicationInfo(User user): String
+ newApplication(User user, String[] appContent): void
+ updateApplication(String[] appContent): void
+ getUserInfo(User user): String
+ getUserStatistics(User user): String[]
```

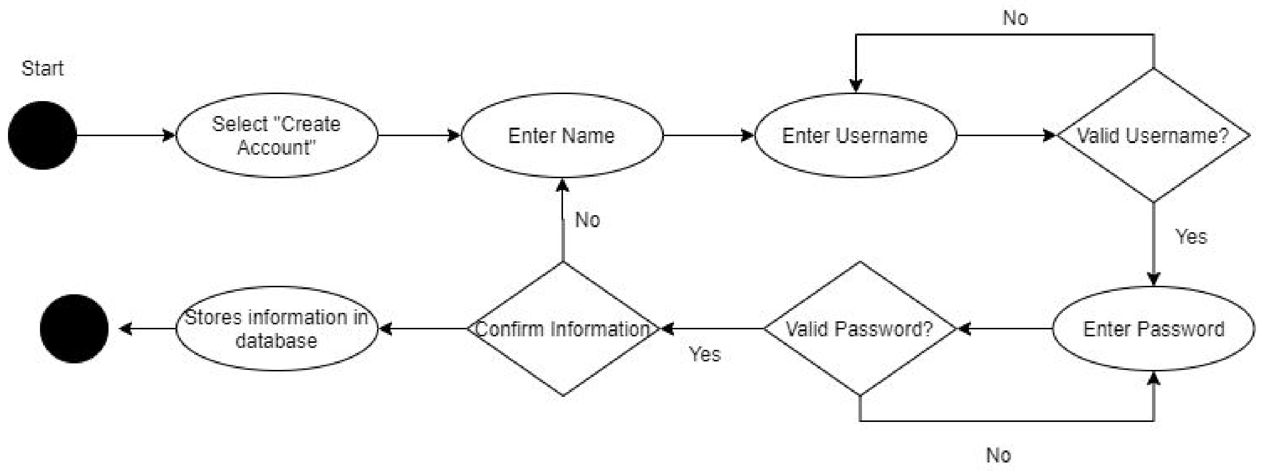
Class Interaction Diagram



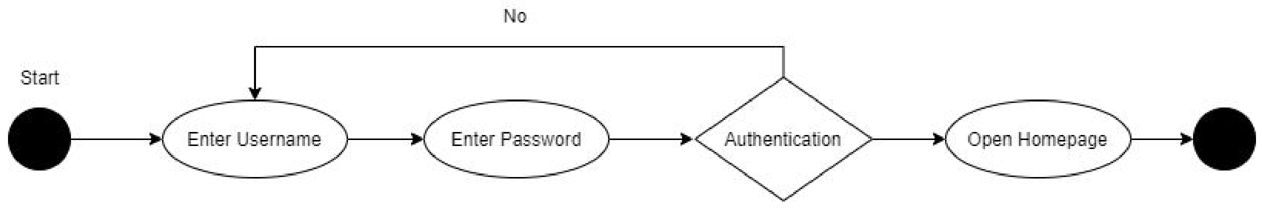
9.2 Dynamic - Behavioral Models

Event Diagrams:

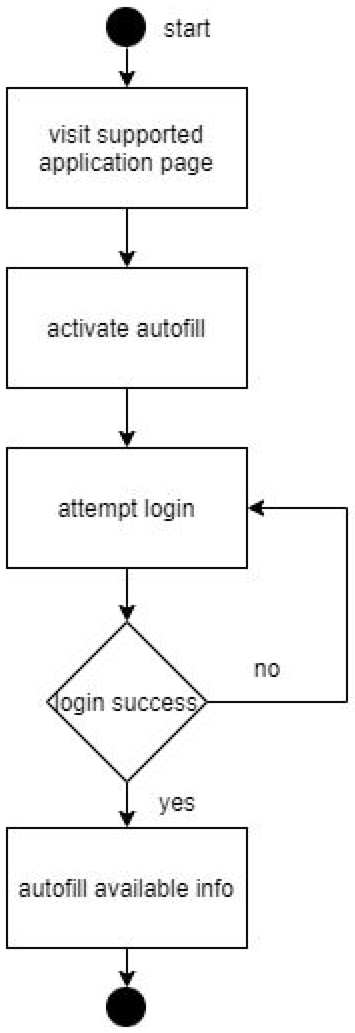
- Create Account



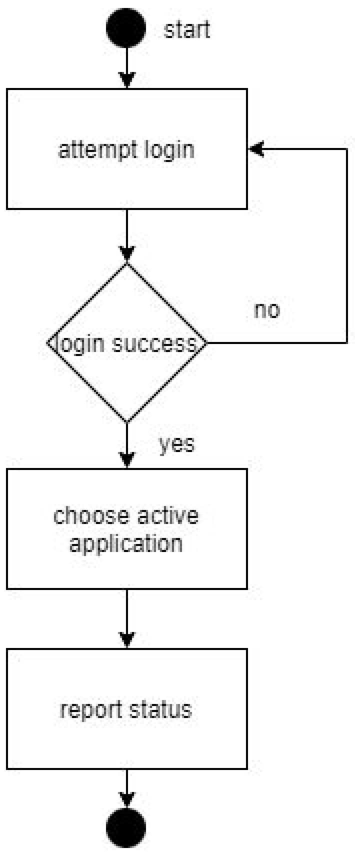
- Login



- Autofill Application

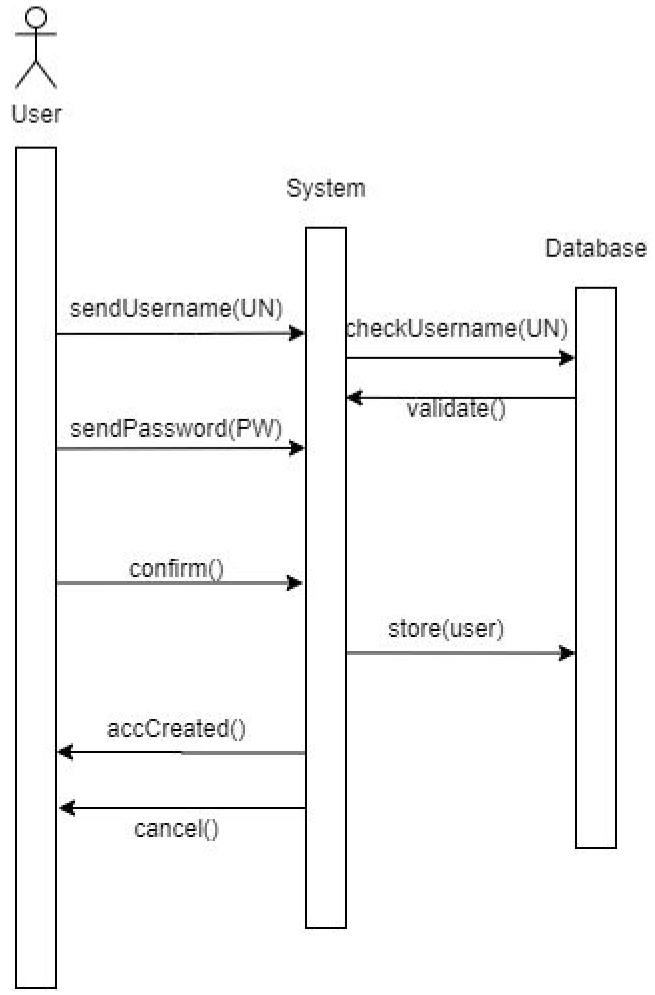


- Update Application Statuses

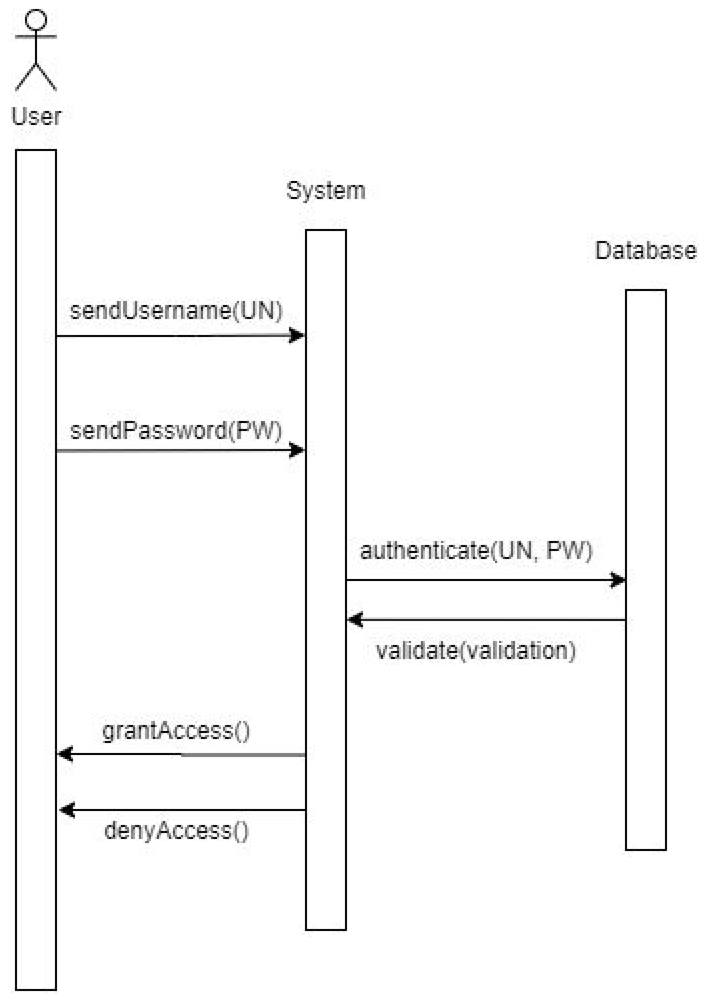


Sequence Diagrams:

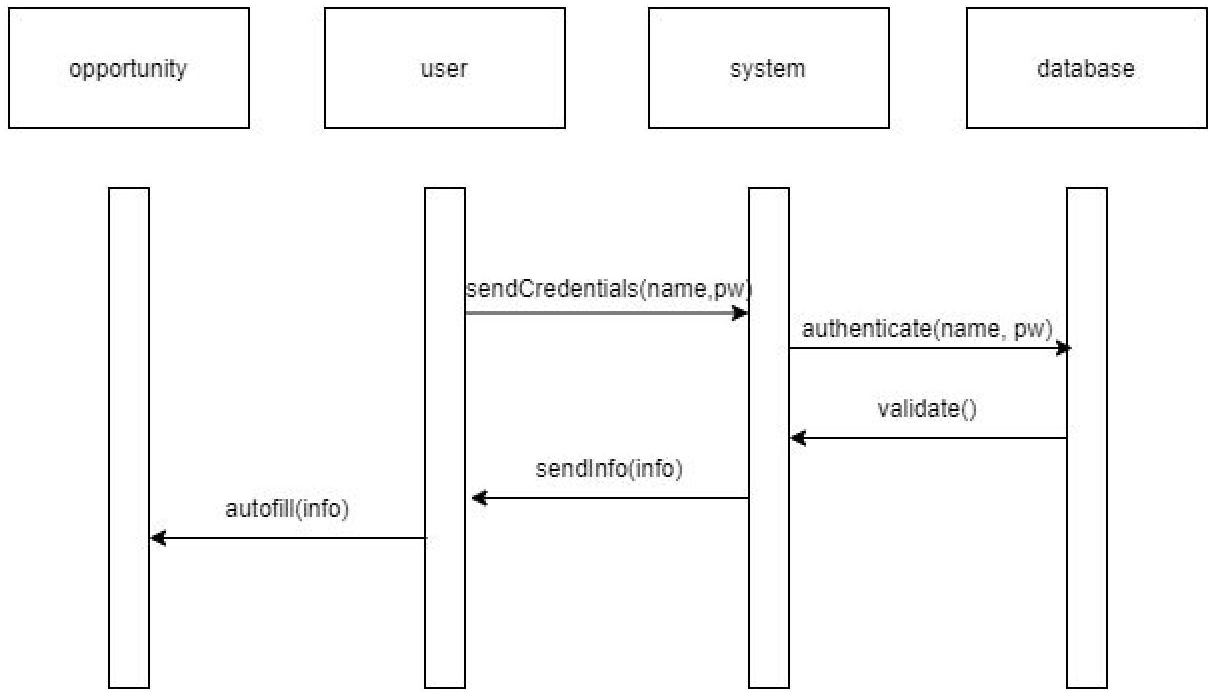
- Create Account



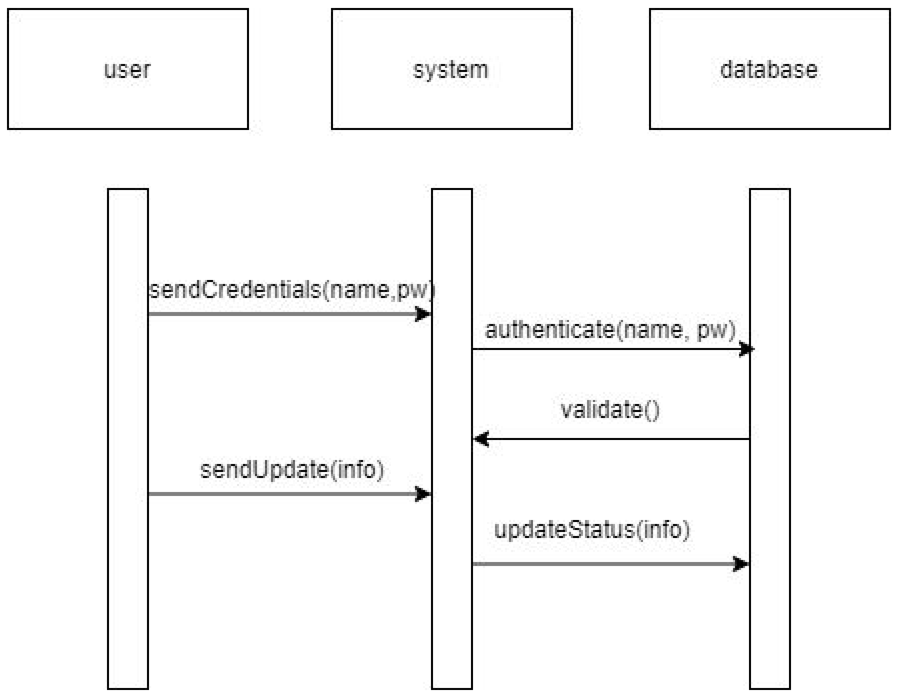
- Login


User
System
Database
sendUsername(UN)
sendPassword(PW)
authenticate(UN, PW)
validate(validation)
grantAccess()
denyAccess()

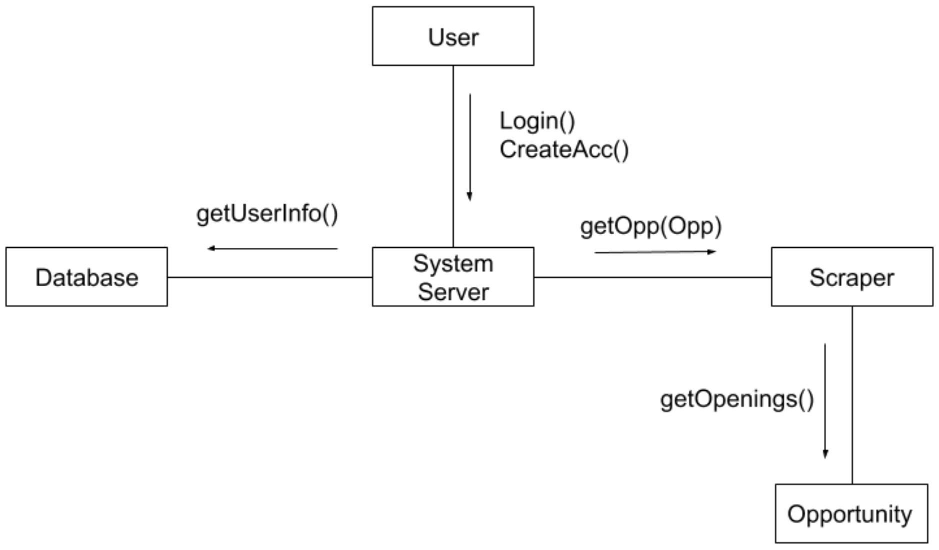
- Autofill Application



- Update Application Statuses



Collaboration Diagram



10. Evolution of the SRS

Legend:
+ ADDED
- REJECTED
* MODIFIED

Version 1.0
+ SRS Cover Page
+ SRS Formatting
+ 1. Document Purpose
+ 2. Introduction
+ 3. Glossary
+ 4. Reference Documents
+ 5. Business Requirements
+ 10. Evolution of the SRS
+ 13. Appendices

Version 1.1
+ SRS Cover Page
+ 6. User Requirements
+ 7. System Architecture
+ 8. Detailed System Requirements - Use Cases
* Project Info - Changed name of project
* Section 1 Introduction - changed scope descriptions
* Version number Page 3
* Section 2.6 Constraints
* Section 4 version number
* Section 13.4 & 13.5 count in hours

Version 1.2
+ Section 9
+ Section 11
+ Section 12
+ Section 14
+ Section 13.1
+ Section 13.2
+ Section 13.3
+ Section 13.6
* Version number of document
* Date of document

11. Rationale

None

12. Notes

None

13. Appendices

13.1 System Test Plan Requirements

The system will be tested by the development team and users through regular usage. Scenarios that occur that will test the system are as follows.

Scenario 1:

Johnny Appleseed is the average 20-year old software engineering student looking for internship opportunities during summer break. He opens the messaging application on his phone and issues a command to the bot to give him the status of all of the applications he's been tracking. The bot tells him that Company A and Company B have both opened applications and gives him the link to apply. Johnny clicks the link and applies then updates the application status for both companies to "applied" with the bot.

Tested:

- Storing information of companies that the user is interested in
- Providing the user with application statuses
- User customizability of the system
- The system giving users notifications
- The system giving users links to applications

Scenario 2:

Two weeks later, Johnny opens the messaging application and runs a command that tells the bot to give him application updates. Company A and B have both rejected Johnny. Dismayed, Johnny deletes both companies from his list and adds Company C to his list. The bot confirms the deletion of Companies A and B and the addition of Company C. Johnny then closes the messaging application. Later in the day, Johnny receives an automatic notification from the bot telling him that the applications for Company C have opened.

Tested:

- Deleting companies from the user's list
- Adding companies to the user's list
- Automatic tracking and notification of a company's application opening by the system

There are no simulators required for testing the system. The system is tested through regular usage.

13.2 Qualification Provisions

During the verification process for this document, the quality of this document is verified. Each section is checked to ensure that they contain the correct material. This is done by comparing what is written to the description of what should be written on the template. In addition, the grammar of the writing in each section is checked to ensure that what is written makes sense and adequately describes the project. To ensure that the document is consistent with itself, each sections' material is reviewed and compared to related material in other sections.

13.3 Requirements Traceability

Forward Traceability:

The basis of the project stems from software engineering internship websites. With the internship websites, data can be scraped about application dates and status. That data is sent to the user via a messaging application bot, which the development team will need to create. The messaging application bot is able to provide the user with messages informing them of application statuses which is the endpoint of the project.

Backward Traceability:

The endpoint of the project is a messaging application bot that is able to interact with the user and their job applications. This bot is created by the development team. The bot is designed to visit and scrape data off of websites including, but not limited to, job openings, application dates, and application statuses. The websites that this information can be found on host software engineering internship opportunities - the basis for the project.

13.4 Schedule Tracking

Artifact or Deliverable	Who	Estimated	Actual	Difference
Initial SRS	Leo Liu	October 7, 2020	October 7, 2020	0 hours
	Rihui Zheng	October 7, 2020	October 7, 2020	0 hours
	Md Abedin	October 7, 2020	October 7, 2020	0 hours
	Kyle Lin	October 7, 2020	October 7, 2020	0 hours
	Summary	October 7, 2020	October 7, 2020	0 hours

Artifact or Deliverable	Who	Estimated	Actual	Difference
Final SRS	Leo Liu	November 19, 2020	November 19, 2020	0 hours
	Rihui Zheng	November 19, 2020	November 19, 2020	0 hours
	Md Abedin	November 19,	November 19,	0 hours

		2020	2020	
	Kyle Lin	November 19, 2020	November 19, 2020	0 hours
	Summary	November 19, 2020	November 19, 2020	0 hours

Cumulative

Who	Estimated	Actual	Difference
Leo Liu	November 19, 2020	November 19, 2020	0 hours
Rihui Zheng	November 19, 2020	November 19, 2020	0 hours
Md Abedin	November 19, 2020	November 19, 2020	0 hours
Kyle Lin	November 19, 2020	November 19, 2020	0 hours

13.5 Defect Tracking

Artifact or Deliverable	Who	Estimated	Actual	Difference
Initial SRS	Leo Liu	0	1	1
	Rihui Zheng	0	0	0
	Md Abedin	0	0	0
	Kyle Lin	0	0	0
	Summary	0	1	1

Artifact or Deliverable	Who	Estimated	Actual	Difference
Final SRS	Leo Liu	0	0	0
	Rihui Zheng	0	1	1
	Md Abedin	0	0	0

	Kyle Lin	0	0	0
	Summary	0	1	1

Cumulative

Who	Estimated	Actual	Difference
Entire Team, v 1.0	0	1	1
Entire Team, v 1.1	0	1	1
Entire Team, v 1.2	0	0	0

13.6 Dictionaries

Class

Name	Description	Method	Attributes
User	Generated by database after successful login, and used for future queries regarding that user	`None`	`username` `loginToken`

Opportunity	Intermediate class generated by classes that implement the Scraper interface.
	Stores information about one opportunity
		`None`	`company`
			`type`
			`position`
			`field`
			`description`
			`requirement`
			`applicationDeadline`

<<interface>> Scraper	Interface for all employer website scrapers	`scrape` `getOpenings`	`url` `openings`

<<interface>> Database Management System	Interface for handling database connection and queries with different DBMS	`validateCredential`
		`createAccount`
		`getOpportunityByType`
		`getOpportunityByCompany`
		`getOpportunityByField`
		`getApplicationInfo`
			`conn`

		newApplication updateApplication getUserInfo getUserStatistics	

Methods

Name	Description	Class	Arguments	Return Type
scrape	Scrape the website at the url for opportunities	<<interface>> Scraper		void
getOpenings	Returns opportunity[] scraped	<<interface>> Scraper		opportunity[]
validate Credential	Authenticates login information	<<interface>> Database Management System		User
createAccount	Create account	<<interface>> Database Management System		User
getOpportunity ByType	Retrieve opportunities on database by type (internship/fulltime/etc)	<<interface>> Database Management System	String type	opportunity[]
getOpportunity ByCompany	Retrieve opportunities on database by employer	<<interface>> Database Management System	String company	opportunity[]
getOpportunity ByField	Retrieve opportunities on database by field (ex: appSec)	<<interface>> Database Management System	String field	opportunity[]
getApplication	Retrieve	<<interface>	User user	String

Info	application by user	> Database Management System		
new Application	Add a new application into the database	<<interface>> Database Management System	User user, String[] appContent	void
update Application	Update an existing application in the database	<<interface>> Database Management System	String[] appContent	void
getUserInfo	Takes in a User object and return that user's info	<<interface>> Database Management System	User user	String
getUser Statistics	Takes in a User object and return that user's application statistics	<<interface>> Database Management System	User user	String[]

Attributes

Name	Description	C/S	Type	Permissions
company	Employer	simple	String	R/W
type	Type of opportunity	simple	String	R/W
position	Offered position	simple	String	R/W
field	Field of the position	simple	String	R/W
description	Description of the opportunity	simple	String	R/W
requirement	Requirements for the opportunity	simple	String	R/W
application Deadline	Application deadline	simple	Date	R/W

username	Username	simple	`String`	`R/W`
token	Login token generated on successful credential validation	complex	`LoginToken`	`X/X`
url	url to be scraped	simple	`String`	`X/X`
openings	List of all openings scraped	complex	`Opportunity[]`	`X/X`
conn	DBMS connection	complex	`Connection`	`X/X`

Relationships

Name	Description	From class	To class	Optional/ mandatory	Cardinality
generates	Scraper generates a list of opportunities	Scraper	Opportunity	mandatory	one to one
selects	User is used to retrieve opportunities	User	Opportunity	mandatory	one to many
updates	User updates their database entries	User	Database	mandatory	many to one
contains	Database contains opportunity entries	Database	Opportunity	mandatory	one to many

Key Events

Name	Description	Motive	Action	Pre-conditions	Post-conditions	State Change

Create Account	A user can create an account through the client interface	Sign up for the app	1. User fills form and clicks "Create Account" button 2. Username checked against database to ensure uniqueness 3. Account created	The user is not already logged in	The user has an associated account in the database, and is logged in.	N/A
Login	A user can log in to their account	Access the app	1. User enters username and password 2. System checks database to authenticate information 3. User gets redirected to the homepage on success	The user is not already logged in and has an account	The user is now logged in.	Logged in
Autofill Job Application	The user will choose a job and the system will fill out fields of the job application form for them automatically	Easily fill out an application	1. User info retrieved from database 2. Info entered into an application form	The user needs to have an account and some of their application info saved in the system	The job application they selected will have some fields filled out	N/A
Update Application	The user reports the	Keeps the user's	1. User selects one	The user heard	The database	N/A

Status	status of their job application for their personal statistical tracking	application status up to date	of their active application 2. Selected application's entry in the database is modified	back from potential employers after applying to an opportunit y	now holds the status of the user's applicatio n.	

14. Index

N/A