
Software Requirements Specification

for

Apprentice

Version 1.0 approved

**Prepared by Kordell Hutchins, Henry, Michael Jarvis, Adesimisola,
and Mark Angelot**

Group DNWS

4/2/22

Table of Contents

Table of Contents	ii
Revision History	ii
1. Introduction	1
1.1 Purpose	1
1.2 Document Conventions	1
1.3 Intended Audience and Reading Suggestions	2
1.4 Product Scope	2
1.5 References	2
2. Overall Description	3
2.1 Product Perspective	3
2.2 Product Functions	3
2.3 User Classes and Characteristics	3
2.4 Operating Environment	3
2.5 Design and Implementation Constraints	3
2.6 User Documentation	3
2.7 Assumptions and Dependencies	4
3. External Interface Requirements	4
3.1 User Interfaces	4
3.2 Hardware Interfaces	5
3.3 Software Interfaces	6
3.4 Communications Interfaces	6
4. System Features	7
5. Other Nonfunctional Requirements	7
5.1 Performance Requirements	7
5.2 Safety Requirements	8
5.3 Security Requirements	8
5.4 Software Quality Attributes	8
6. Other Requirements	9
7. User Stories	9
Appendix A: Glossary	10
Appendix B: To Be Determined List	10

Revision History

Name	Date	Reason For Changes	Version
TBD	TBD	TBD	TBD

1. Introduction

1.1 Purpose

The product of this software requirement is Apprentice. The release number is 1.0. The scope of the product covered is the basic framework functionality and backend functionality. Scope will include:

- Registration: interaction primarily backend to register their account on the platform.
- Login: interaction primarily backend to login their account on the platform.
- About us: information describing what the project is about.

1.2 Document Conventions

- Bolding signals the heading or specifications for a certain demographic of readers.
- Highlighting in yellow describes important requirements.

1.3 Intended Audience and Reading Suggestions

The intended audience for this document is for:

- Developers
- project managers
- document writers

The SRS is organized as:

1. Overall Description - Contains a description of what the SRS document is covering and other general information regarding the requirements of the project.
2. External Interface Requirements - Describes the requirements of external interfaces like Frontend aspects (Colors, font size, web page structure, etc..), Backend aspects (Flow of nginx with Web Storage API, MySQL integration with Javascript, etc...)
3. System Features - Describes the features that the platform will have in the Frontend (Changing Username, Profile theme, Liking post, accessing custom feed, etc...), Backend (Create-Read-Update-Delete in database, HTTPS certs with nginx, Web Storage API for accounts and other customization technologies, etc...)
4. Other Nonfunctional requirements - Describing other requirements like performance (Balancing load of small user base, etc...), safety(establishing user timeout features, etc...) security (use HTTPS, etc...), and software quality attributes(have a modern flow of the platform).
5. Other requirements - Describes other requirements that weren't covered in the external and other nonfunctional requirements.
6. User stories - Stories authored by clients/users that will be using our platform.

For **developers** interpreting the document, read the SRS document in this sequence:

1. Overall Description
2. System Features
3. User Stories
4. External Interface Requirements

5. Other Nonfunctional requirements
6. Other requirements

For **project managers** interpreting the document, read the SRS document in this sequence:

1. Overall Description
2. User Stories
3. External Interface Requirements
4. Other Nonfunctional requirements
5. Other requirements
6. System Features

For **document writers** interpreting the document, read the SRS document in this sequence:

1. System Features
2. Overall Description
3. External Interfaces Requirements
4. Other nonfunctional requirements
5. Other requirements
6. User Stories

1.4 Product Scope

Apprentice is a collaboration platform for students that serves as a platform that provides information about deals/hidden features associated with their student account. There isn't a platform out there that organizes information to tell students specifically what deals are available and associated with their student account. More information regarding benefits and goals are in the Group DNWS scope and vision document.

The boundaries for this project are, in the included:

- Building platform that is hosted on a single linux server.
- Building a backend that manages the creation of user accounts and Posts.
- Building a backend that hosts the site that is secure for public access.
- Building a backend that uses a modern form of user sign in and verification.
- Building a frontend that establishes a modern UI
- Building a frontend that establishes a modern flow with web interaction.
- Building a frontend that organizes posts into categories and feeds.
- Building a frontend that allows users with custom profile themes and other addons.

In the excluded:

- Building AI to moderate chat engagement on the platform.
- Building Tier platform for a subscription service.
- Building more customization capabilities with the subscription service.
- Building a MYSQL database to handle multiple server integration.
- Building Load Balance capabilities to manage user traffic.

1.5 References

Other external documents are Group DNWS Scope and Vision and Group DNWS Budget Development

2. Overall Description

2.1 Product Perspective

The program is designed to innovate the current systems used by students as well as product vendors such as books, food etc. Students will be provided a platform that allows them to share hidden features/deals other students may not know are available on or nearby their institution. These reviews are solely for students as it will be associated with their student accounts.

2.2 Product Functions

The main function of the product is to allow students to share hidden features/deals associated with their student account that other students may/may not know. It also allows businesses around the campus to be further exposed if they are not well known but give quality service at a student's budget

2.3 User Classes and Characteristics

The main user class would be students as having a student account is required to use this product as well as credible business owners that would like to advertise their business. The most important user class is the student as they would be looking for information based on hidden deals around their campus.

2.4 Operating Environment

The operating environment would be a web browser particularly the latest version of google chrome, it will also be provided as an extension to google chrome once students log in to their respective student emails.

2.5 Design and Implementation Constraints

Some constraints that may/may not be of issue is privacy of users when they post their reviews. Some end users may be concerned their information may be leaked as we are requiring student information in order to fulfill product goals. Other constraints would be an overload of information that may cause the system to crash, if we have for example more users/reviews than our memory could handle it may result in a shutdown.

2.6 User Documentation

For the documentation of this project there will be a manual that includes the backend and the front end of the project. It will include all the tools and languages that are used to build the software from scratch.

2.7 Assumptions and Dependencies

It is assumed that there would be a few product development problems as this is an innovative app. Those problems include constraints from third party components. As this would be a web application associated with student accounts, we may run into privacy issues as we need the student information to verify whether or not they are indeed students. In addition to working with various schools and their web browsers, it is known some schools use blackboard and others use canvas. Therefore translating the software to various school browsers may pose an issue.

3. External Interface Requirements

3.1 User Interfaces

Apprentice will be a web based application; below is a model of how the login page will look. When a user changes the tab, it will be indicated in a different color. Below “Login” changes to yellow. The font used throughout the web pages will be Arial. The color scheme will be changed to black, gold, and white, for the Towson colors.

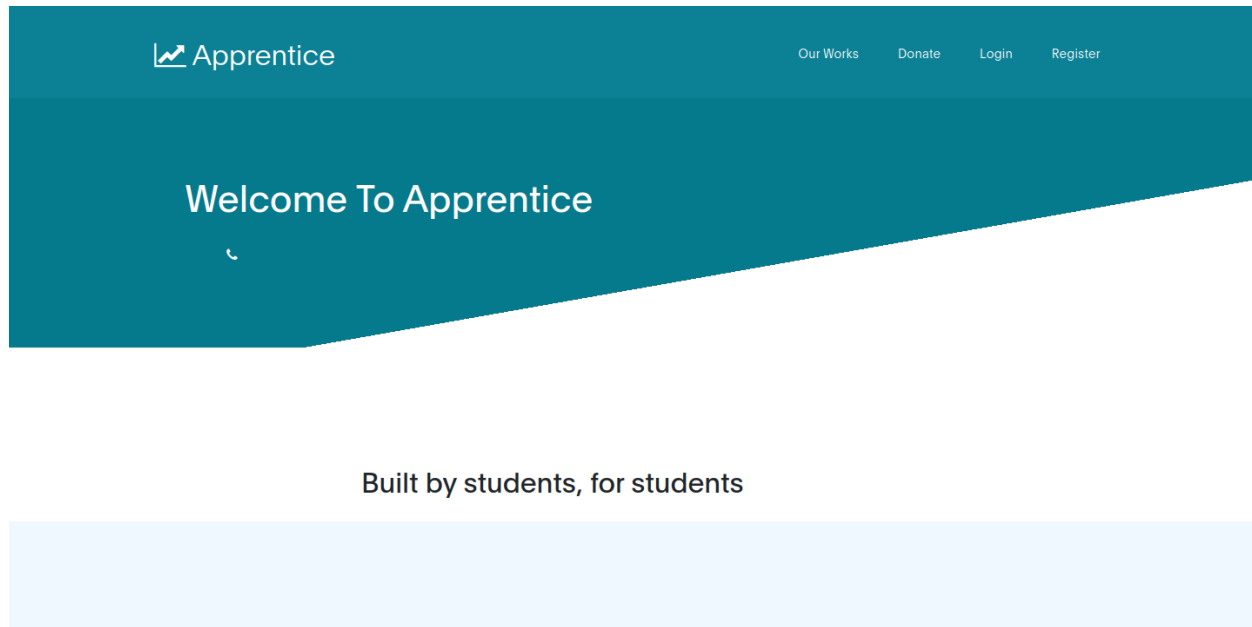


Login to interact with posts
and make your own

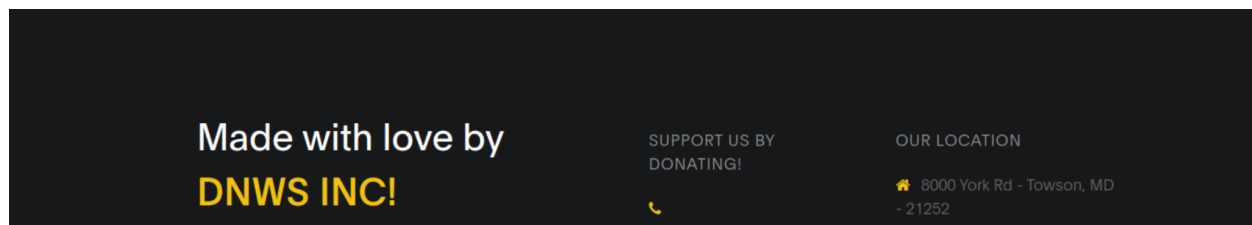
Username:

Password:

Apprentice will have a home page that describes the purpose and mission of the website, as seen below. This is a skeleton of what the page will look like.



We will also include a footer to all pages on the website, which will include information about the makers of DNWS and contact information. Below is an example, not final.



There will also be a page which people can post and find information about their student account that they may not know about.

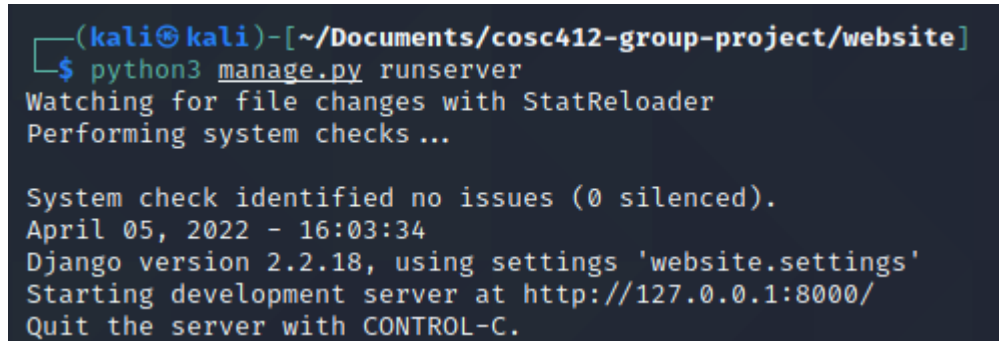
3.2 Hardware Interfaces

Apprentice will be hosted on an Ubuntu web server using the Django web framework. This makes deployment easier and faster. The app will be accessible over https, so it will work with any device that can access the Ubuntu server.

Apprentice can also be hosted on AWS so we don't need to keep our development servers running it, this makes networking easier from our end.

3.3 Software Interfaces

Apprentice uses the Django framework to make connections between frontend and backend, as well as make our WebApp more secure. Django was chosen because many other social media apps, such as Instagram are based on it. It is also based on python, which we prefer over PHP. The database can either be stored on our development server, or it could be hosted by AWS. Below is a picture of how the development server will be started.

A terminal window with a dark background and light-colored text. The prompt is '(kali@kali)-[~/Documents/cosc412-group-project/website]'. The user enters '\$ python3 manage.py runserver'. The output shows 'Watching for file changes with StatReloader', 'Performing system checks ...', 'System check identified no issues (0 silenced).', the date and time 'April 05, 2022 - 16:03:34', 'Django version 2.2.18, using settings \'website.settings\'', 'Starting development server at http://127.0.0.1:8000/', and 'Quit the server with CONTROL-C.'

```
(kali@kali)-[~/Documents/cosc412-group-project/website]
$ python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks ...

System check identified no issues (0 silenced).
April 05, 2022 - 16:03:34
Django version 2.2.18, using settings 'website.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

3.4 Communication Interfaces

Since Apprentice is a WebApp, it will need connections over the internet. The development server will be run over HTTPS because users will need to be able to access it from other devices. In deployment, it will be run through AWS, which has even more security precautions. The Apprentice code will have self protections from SQL injection, Cross-Site Request Forgery, and Cross-Site Scripting attacks. We will also hash all passwords to make password attacks harder.

4. System Features

Req #	Features/ Function	Functional Requirement Statement	Priority (L, M, H)
FR#1	Access Authentication	The system will allow users to create an account profile using a provided link.	H
FR#2	Access Authentication	The system must allow users to delete their account profile.	H
FR#3	Access Authentication	The system must allow users to retrieve their username.	H
FR#4	Access Authentication	The system must allow users to reset their password.	H
FR#5	Access Authentication	The system must allow users to log out of the system.	H
FR#6	Profile Management	The system must allow users to modify their account information.	H
FR#7	Profile Management	The system must allow users to access their account profile information while offline.	M
FR#8	Appointment Management	The system must allow users to create an appointment.	H
FR#9	Appointment Management	The system must allow users to modify appointment information.	H
FR#10	Appointment Management	The system must allow users to cancel appointments.	H

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Req #	Technical Category	Non-Functional Requirement Statement	Priority (L, M, H)
PR#1	Performance	The webpage must load within no more than 5 seconds for users that access the website using a web browser.	H
PR#2	Performance	The mobile app must load within no more than 10 seconds for users that access the website using a mobile connection.	H
PR#3	Performance	New updates must not interfere with older features. Webpages should be loosely coupled	H
PR#4	Performance	Forum data should be kept in a different database from data from user posts	M
PR#5	Performance	The system has to run from a web server	M
PR#6	Performance	The performance shall depend upon hardware components of the client/customer.	L

5.2 Safety Requirements

Req #	Technical Category	Non-Functional Requirement Statement	Priority (L, M, H)
SR#1	Safety	The system will ensure that all data provided by the user should not be susceptible to unauthorized changes	H
SR#2	Safety	User private data must be hashed and not visible to other users	H

5.3 Security Requirements

Req #	Technical Category	Non-Functional Requirement Statement	Priority (L, M, H)
SC#1	Security	The system will have authentication to avoid unauthorized access	H
SC#2	Security	The system will support two factor authentication	M
SC#3	Security	The system shall support cloud based authentication	L
SC#4	Security	The system will hash user credentials before storing them in a database	H
SC#5	Security	The system will ensure that access to the database access is limited to only be accessed by other system when necessary	L
SC#6	Security	The system will implement a password strength requirement. It checks that the user password length is 8 (letters, characters and numbers) and is not susceptible to dictionary attacks.	H
SC#7	Security	The system shall automatically log out all customers after a period of inactivity.	M

5.4 Software Quality Attributes

Req #	Technical Category	Non-Functional Requirement Statement	Priority (L, M, H)
SQ#1	Scalability	The system will be able to scale enough to support 200,000 concurrent users	H
SQ#2	Availability	The system should apply load balancing to minimize the impact of potential system failures	H
SQ#3	Maintainability	The system will be loosely coupled to reduce interdependence and allow for quick updates	M
SQ#4	Reliability	The database update process must roll back all related updates when any update fails	M
SQ#5	Availability	The system's database will copy all content into a replicate database in case of database failure	H
SQ#5	Compatibility	The application must support iOS devices running iOS 8 and above as well as android devices run android 5.0 and above	L

SQ#6	Supportability	The source code developed for this system shall be maintained in configuration management tool	M
SQ#7	Useability	The system shall provide use of icons and toolbars.	M

6. Other Requirements

Req #	Technical Category	Non-Functional Requirement Statement	Priority (L, M, H)
SQ#1	Safety	The system will be able to scale enough to support 200,000 concurrent users	H

7. User Stories

USE R STORY ID	ROLE		FUNCTION PERFORMED			BUSINESS REASON
US#1	As a	Student	I need the system to	post deals	so that I may	share deals with other students
US#2	As a	student	I need the system to	organize information about new features	so that I may	learn those features in my custom feed
US#3	As a	student	I need the system to	have user customization	so that I may	customize my profile to my liking
US#4	As a	student	I need the system to	Add comments to posts	so that I may	further ask questions or ask for clarification.
US#5	As a	student	I need the system to	Like posts	so that I may	like posts of deals or comments to show my engagement.

Appendix A: Glossary

AWS: Amazon Web Service

HTTPS: Hypertext Transfer Protocol Secure

Appendix B: To Be Determined List