**CS673 Software Engineering** 

**Team 4 - Rxcellent**

**Project Proposal and Planning**

| Team Member | Role(s) | Signature | Date |
| --- | --- | --- | --- |
| Zahit | QA Leader | *Zahit Odabas* | 10-18-2022 |
| Nivan | Implementation Leader | *Nivan Hanjura* | 10-19-2022 |
| Ryan | Team Leader | *Ryan Burns* | 10/18/2022 |
| Chenfei | Design Leader | *Chenfei Yu* | 10/19/2022 |
| Ignacio Moral | Requirement Leader and Configuration Leader | *Ignacio Joaquin Moral* | 10-19-2022 |
| Tsing | Security Leader | *Youqing Shao* | 10-19-2022 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Revision history**

| **Version** | **Author** | **Date** | **Change** |
| --- | --- | --- | --- |
| **1.01** | **Ryan Burns** | **9/25/2022** | **7:20 PM** |
| **1.02** | **Ryan Burns** | **9/30/2022** | **9:00 PM** |
| **1.03** | **Zahit Odabas** | **10/2/2022** | **12:50 PM** |
| **1.04** | **Youqing Shao(Tsing)** | **10/9/2022** | **20:11PM** |
| **1.05** | **Ignacio Joaquin Moral** | **10/11/2022** | **6:10 PM** |

| **1.06** | **Ryan** | **10/14/2022** | **4:10 PM** |
| --- | --- | --- | --- |

[Overview](#_pkxrs8xta240)

[Related Work](#_rk0k1bxm1b7v)

[Proposed High Level Requirements](#_y18wby8x0gqp)

[Management Plan](#_yj0es8ywxklr)

[Objectives and Priorities](#_nxeeppkjxgn4)

[Risk Management (need to be updated constantly)](#_tk7yixobah8p)

[Timeline (need to be updated at the end of each iteration)](#_iksrndohvx29)

[Configuration Management Plan](#_y2df6gt75z6a)

[Tools](#_dzly5b9kz982)

[Deployment Plan if applicable](#_sd8zu6r3jisd)

[Quality Assurance Plan](#_g2d99tl72g0f)

[Metrics](#_vwjduhc9wuah)

[Code Review Process](#_hx3eaiwb8v3m)

[Testing](#_l9xnpmd6hh0y)

[Defect Management](#_5amsh8h9f0c7)

[References](#_g0ui1b9fwixo)

[Glossary](#_up42vgk6nn9b)

# 

# Overview

We’re building an e-commerce pharmacy website called Rxcellent that sells prescription medications to patients who want the convenience to refill their prescription with a click of a button rather than calling the pharmacy or going to the pharmacy itself.

The basic functionality of the proposed software system would be that a patient will go to a doctor’s visit. Then the doctor/nurse will write a prescription for him using our Rxcellent Prescription portal. When a prescription is entered into the system, a unique prescription number will be generated. This number will be stored in our system along with the prescription and at the same time, it will be sent to the patient’s email address that is given in his prescription. After that the patient will be able to use the Prescription Search form in our system by entering his name, date of birth and prescription number. When he query his prescription, he will be able to order his prescription to his desired address. Another functionality of our system is that a user can buy over the counter medicine without login. OTC medicines will be listed in Rxcellent website and can be added to the shopping cart. Possible technology stack to be used - JS, React, MongoDB, Firebase, Express & Node.

# Related Work

* Amazon Pharmacy - provides a web-based marketplace for prescriptions with the market price. After selecting a needed medication , a customer can then add their insurance to check the price with insurance and/or they can transfer their Rx to Amazon Pharmacy (by clicking a Big Yellow Button). **What are the differences between our software and this software?** We don’t require a log in to get a prescription while other apps do.
* Mark Cuban CostPlus Drug Company **What are the differences between our software and this software?** We don’t require a log in to get a prescription while other apps do.
* **Walmart Pharmacy** - This pharmacy offers like other pharmacies prescription refills and transfers online. If you already have a prescription you can request a refill or create an account, and request a Rx transfer. After a customer request is made and goes through the approval process the customer is given an option to pick it up at a local Walmart pharmacy. The customer can choose a pharmacy to pick up their Rx at a location within a 20 mile radius of the customer’s location.

# Proposed High Level Requirements

## Functional Requirements:

* + *Essential Features*
    1. Order Rx Online:
       1. As a person with a prescription, I want to be able to order prescription medicine online so that I don’t need to walk to the store every time.
       2. As a customer, I want to be able to order over the counter medicine online so that I don’t need to walk to the store every time.
    2. Order non-Rx medication online:
       1. As a customer, I want to search OTC medicine by keywords or active ingredients so that I find the medicine easily
    3. Manage Shopping Cart:
       1. As a customer, I want to be able to add/remove medicine to my cart, so that I can check out multiple medicines at once.
    4. Get Rx Without a Login:
       1. As a customer, I don’t want to register for an online pharmacy to be able to order medicine, so that I won’t give my personal information and I don’t have to manage my password
    5. Doctor/Prescriber Login:
       1. As a prescriber, I want to be able to log into prescription panel so that I can write e-prescriptions
    6. Doctor Should Issue Rx:
       1. As a prescriber, I want to be able to write e-prescriptions, so that patients can order their prescriptions by using it.
  + *Desirable Features*

1. Customer Payment Method:
   1. As a customer, I want to pay for my medicine with a card so that I can complete my transaction and get my medicine delivered to my address.
2. Customer Can Have Account:
   1. As a user, I would like to have an option to create an account so that I can track my prescriptions and order history.
   * *Optional Features*
     1. Inventory Control:
        1. As a pharmacy store manager, I want to be able to update medicine stocks on the web store, so that the web store always presents the actual stock.

## Nonfunctional Requirements:

* + Web-based
  + Javascript-Based Application
  + High-level security
  + MongoDB Database
  + Express
  + Online Hosting
  + Accessible on mobile devices
  + Unique Branding
  + Material UI (MUI)
  + Security Requirements:

## 1: developing

* web
  + When users register, they need to agree privacy agreement
  + All sensitive can’t show the url or page
  + Prevent evil scripts inject into project
  + Use cookie to save infos before getting acceptance of users
* Back-end
  + The secret key saving refers to important information such as the secret key stored by the object and the public key encrypted and decrypted by the system interface
* Source
  + Project website need a filing Number, like © Copyright 1999-2022 CVS.com
  + All resource(fonts, picture, package……) can’t be infringed
  + Website’s url use “https” rather than “http”.
* Coding
  + No bugs, no code leakage
  + We must do more than unit tests and two case tests before release.
  + Use typescript for stricter data isolation and interface definition
* Security misconfiguration

Security configuration errors can occur at any level of an application stack, including platforms, web servers, application servers, databases, architectures, and custom code. Attackers can gain authorized access to the system by accessing default accounts, unused web pages, vulnerabilities of uninstalled patches, unprotected files and directories, etc

How to prevent:

* Automated installation and deployment
* Timely understand and deploy software updates and patches for each link
* Implement vulnerability scanning and security audit

## 2: interface

* All interface need authentication processing
  + equest data or response data need encrypted(phone, email…)
  + Use jwt to identify user’s information
* Escape ESCAPE for special characters entered(like '%M%' ESCAPE ‘M’)
* Prevent interface header was modified

## 3: attack

* Prevent violent login
* Web need to prevent XSS (Cross Site Scripting)
* Web need to prevent CSRF（Cross-site request forgery）

## 4: database

* Make sure users’ informations are safe
* Prevent SQL injection attacks
* Don’t delete all table or data, just add,edit and search

# Management Plan

## Objectives and Priorities (Basically your project goal)

(Please describe your project objectives with highest priority first. Project Goals can include but are not limited to complete all proposed (essential) features, deploy the software successfully, the software has no known bugs, maintain high quality, etc).

Satisfy all essential features and deploy the software successfully with little to no bugs, and maintain high quality..

## Risk Management (need to be updated constantly)

(Please write a summary paragraph about the main risks your group identified and how you plan to manage these risks. Then use the separate google sheet for detailed risk management. The template is provided in the same folder with this file. Please provide the link to the sheet.)

Risk management - If a team member drops the class we will have an emergency meeting to address any shortfalls the loss will cause on Discord. If the MUI doesn’t fit our needs we are aware of other tools that could be implemented. The team has been encouraged to take backup roles to mitigate any potential roadblocks during development.

**Risk Management Sheet Link:**

## Timeline (this section should be filled in iteration 0 and updated at the end of each later iteration)

| Iteration | Functional Requirements(Essential/Desirable/Option) | Tasks (Cross requirements tasks) | Estimated/real person hours |
| --- | --- | --- | --- |
| 1 | Create a homepage for the User / provide a form for prescription submission | Define requirements, consider design structure,  Complete a design  prototype | 56 hrs per person |
| 2 | User should submit a form and upload a file | Integration tests will begin | 56 hrs per person |
| 3 | The program should be complete where the user can search select and upload their prescription | All tests should be passed and there should be no critical bugs | 56 hrs per person |

# 

# Configuration Management Plan

## Tools

(In this project, we will use Git and Github as the version control tools. Please also specify any other tools to be used, e.g. IDE tools, CI/CD tools, container tools, SAST or DAST tools, and any other DevOps tools)

We are going to use VS Code as IDE; Husky for pre-commit lint check; Vercel as the first choice for deployment and Heroku as a backup.

* 1. **Code Commit Guideline and Git Branching Strategy**

When someone is assigned for a developing task, he first creates an issue for it, then creates a branch with the issue id and names it as *name/issue/:id.* We will use commitizen to normalize our commit messages. Messages are divided to following types:

* feat: a new feature
* fix: a bug fix
* docs: documentation only changes
* style: changes that do not affect the meaning of code
* test: adding tests
* chore: changes to the build process or auxiliary tools and libraries

After his development is finished, he pulls a request for merging this branch to development branch, the implementation leader or QA leader will review the code. When merge is done, the related issue will be closed. This issue mechanism is applied to cases of bug fix. After all development is finished, we will do test on test branch, and use main branch for release after all tests are passed.

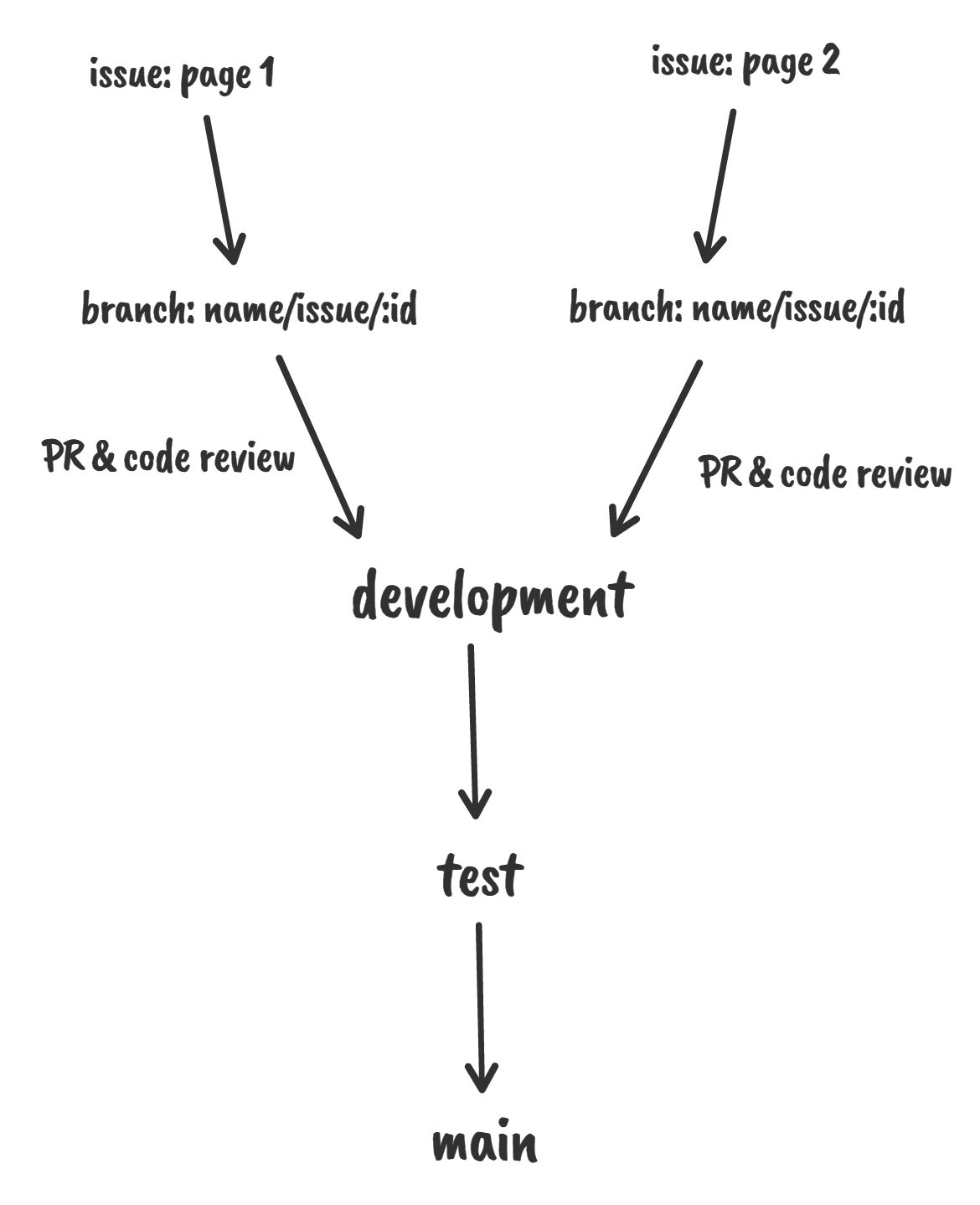
## Deployment Plan if applicable

(If you plan to deploy your application (e.g. your web application), briefly describe how you plan to deploy your application).

Vercel will be our deployment of choice, Heroku will be our backup (maybe AWS if that doesn’t work)

# Quality Assurance Plan

## Metrics

(Describe the metrics to be used in the project to measure the quality of your software. Each metric should be measurable and quantifiable. Examples of metrics include product complexity (LOC, # of files, # of classes, # methods, cyclomatic complexity, etc.) , defect rate (# of defect per KLOC), # of test cases, test case pass rate, cost (# of person hours used), # of user stories completed, etc. **The result of these metrics should be reported in the progress report/ iteration summary sheet.**)

| Metric Name | Description |
| --- | --- |
| # of test cases | Total of unit and integration tests. At least one test case for each method |
| test cases pass rate | We aim a %100 passing rate |
| # of resolved bugs | We aim to identify the bugs with ESLint and resolve all the bugs identified |
| # of user stories completed | We aim to complete all user stories related with essential features |
| Test coverage rate | We aim to achieve at least %75 coverage (using Jest) |
| # of critical bugs | We aim to perform static coding analysis with DeepScan.io and to produce 0 critical bugs |

* 1. Coding Standard

We will use Prettier(VSCode plugin) + ESLint as a JS coding standard. We may tweak the ESLint rules per our needs.

## Code Review Process

(Everyone should review all documents to be submitted. Here you will mainly describe how the code review will be done. Who will review the code, e.g. design or implementation leader will review all code or team members review each other’s code. Do you use pull requests for the code review? Is there a checklist to help review? What feedback should the reviewer provide?)

Design, Implementation or QA leader will review the code. We will use pull requests for code reviews as well as some tools to standardize commit messages ([commitizen - npm](https://www.npmjs.com/package/commitizen)) and automatize pre-commit testing ([Husky - Git hooks (typicode.github.io)](https://typicode.github.io/husky/#/)) to ease code reviewing.

## Testing

(Both manual testing and automated testing should be considered. Both unit testing and integration testing should be considered. Briefly describe the testing tools/framework to be used, the personnel involved (e.g. the QA leader will focus on the integration testing and each developer will unit test their own code), when and what types of testing will be performed, the testing objectives, etc)

Manual testing and unit testing will be performed by all developers during development and before committing (using React-testing-library). Automated tests will be performed at pre-commit stage (using ESLint + Husky).

Another manual testing will be performed by the QA leader after merging into the develop branch. And integration tests will be written by the QA leader. All tests will be run using Jest. Critical bug testing will be performed on the test branch by DeepScan.io

## Defect Management

(Describe the tool to be used to manage the defect (e.g github issues). The types of defects to look at. The actions or personnel for defect management. )

We will mainly use Github as a defect management tool. (For mostly bugs, flaws and failed tests). Anyone can report a defect but mostly the Design leader, Implementation leader or QA leader will report a defect and assign it to the related person.

# References

(For more details, please refer to the encounter example in the book or the software version of the documents posted on blackboard. )

# Glossary

(Any acronym used in the document should be explained here)

Rx: Prescription