Group Number: sdmay20-26
Client and Advisor: Myra Cohen
Team Members/Roles: Matthew Smith – Meeting Facilitator,
Garrett Harkness – Lead Engineer, Shiwei Wang – Test Engineer,
Kevan Patel – Report Manager, Qiwei Li - Scribe
Team Email Address: sdmay20-26@iastate.edu
Team Website: https://sdmay20-26.sd.ece.iastate.edu

Revised: Version 2

Executive Summary

Assurance Recipes Application

DESIGN DOCUMENT

Development Standards & Practices Used

In our group and with our faculty advisor, we decided to work under an agile development process. We all meet as a group once a week to work on different things we need to do for the week, as well as another meeting with our advisor once a week as well. 1-2 weekly sprints are created, with 2 major milestones in each semester. Using Trello, we plan on keeping track of stories, hours, and progress throughout the semester. We also want to use Gitlab for our version control so that we can update our latest code and keep track of commits and pushes. All communication is used via slack and email for Team and Advisor communication. We also want to run under a TDD (Test Driven Development) environment. This improves our code quality and reduces bugs. It also reduces the time spent on rework. We also want to have effective teamwork among our group. We want to allow optimal productivity in a limited amount of time. We also can get differing perspectives and feedback from each other. Good documentation is also extremely key when developing. Good open API documentation allows the project to be handed off to future developers, so they have an easy time understanding the code.

Summary of Requirements

- Functional Requirements:
    - Creation and editing of safety cases.
    - Templated safety cases.
    - Free text.
    - Description page of safety cases.
    - Data safe.
    - Saving of safety cases.
- Nonfunctional Requirements:
    - Performance
    - Scalability
    - Security
    - Reliability
    - Usability
    - Availability

Applicable Courses from Iowa State University Curriculum

- Computer Science 309 – Software Development Practices: A practical intro to managing software development. Process models, requirements analysis, structured and object-oriented design, coding, testing, maintenance, cost and schedule estimation, metrics. Programming projects.
- Computer Science 319 – Construction of User Interfaces: Overview of user interface design. Evaluation and testing of user interfaces. Review of principles of object orientation, object-oriented design and analysis using UML in the context of user interface design. Developing Web and Windows-based user-interfaces.
- Computer Science 363 – Introduction to Database Management Systems: Relational, object-oriented, semi structured and query languages. SQL, XML, and NO-SQL. Database design using entity-relationship model, data dependencies, and relational database design. Application development in SQL-like languages and general-purpose host languages. Web application development. Programming projects.

New Skills/Knowledge acquired that was not taught in courses

Working with a faculty advisor and client face to face is one of the major differences not learned in other courses. Senior Design allows you have a bit more freedom in how you want to design your project and how you want to communicate with your clients. This freedom gives you a free reign of any technology, software, development practices, etc. which allows us to really create the projects the way we want to within the client's limitations. Many of the skills learned in this course may not be on a technical level, other than possibly learning some new languages, but more learning things on a one on one communication level with our client and advisor. We are able to apply skills learned in courses we have taken in the past, such as generating requirements by working closely with our client/advisor, as well as gaining practice with the utilizing an Agile Development process.

Table of Contents

List of figures/tables/symbols/definitions (This should be the similar to the project plan)

1 Introduction

1. Acknowledgement

We would like to acknowledge Myra Cohen for being our faculty advisor and our client during this entire process. She has been an extremely huge help in guiding us in our project and helping us come up with how we want to implement this project.

2. Problem and Project Statement

Our project is the Assurance Recipes application. These recipes involve the use of synthetic biology to genetically modify organisms like E. Coli to help daily routines. Many people want to be certain that this new and exciting opportunity will be safe for both them and the community. That is where assurance cases come in. These cases are a design strategy that helps biology students map out their safety in a notation called Goal Structuring Notation. They ensure the safety of various parts of their experiments and map out their design structure in an efficient way. Our project will allow students and experts to create assurance cases easily and efficiently on an all new website/app and will help students out with how to create these assurance recipes. We want the application to be easily editable where users can create assurance cases easily. With a clean and highly functional UI, being able to edit and create assurance cases is the biggest thing we want to accomplish in this project. There is already a website created that accomplishes this, but the client wants us to make a better and more secure application for Iowa State biology students to easily use.

3. Operational Environment

After discussion with our client/advisor, we have decided that we will develop an Electron application, which utilizes HTML, CSS, and JavaScript to run as a desktop application on the users' computer.

4. Requirements

- Functional Requirements:
    - Creation and editing of safety cases: Users should be able to freely edit and layout their safety cases the way they want to.
    - Templated safety cases: Users should be able to choose from a list of templates since safety cases can be quite complex.
    - Free text: Users should be able to type free text in each diagram creation.
    - Description page of safety cases: We want a description page to describe what assurance recipes and cases are. We also want to describe the importance of them and how they are used in synthetic biology.
    - Data safe: User's data should be stored safely.

    - Saving of safety cases: Users should be able to save their cases on their local machines.

- Nonfunctional Requirements:
    - Performance:
        - Users can run on multiple machines.
        - Short response time.
    - Scalability:
        - Users are able to add assurance cases.
    - Security:
        - Users, program's data, and ports are protected.
        - Follows EU GDPR regulation.
    - Reliability:
        - Provide a stable, safe software.
    - Usability:
        - An easy interface for users to learn and understand: Effective, intuitiveness, low perceived workload.
    - Availability:
        - Testability
        - Manpower
        - Detailed diagnostic procedures.

5. Intended Users and Uses

The intended users for this project are biology students who plan on making assurance recipes in their synthetic biology courses. The application will allow them to easily create these diagrams on their own and from templates so that they can make sure they are running safe and well-designed experiments.

6. Assumptions and Limitations

Our project is planned to be widely used by anyone that wants and needs to use it. It will mostly be used by Iowa State biology students that are performing these experiments in the real world for

their class. We still have discussion on how we want to implement security, but we don t even want any kind of user authentication or important information being stored other than the assurance diagrams being created. We plan to eventually expand this project beyond just Iowa State to be used by Synthetic Biologists.

7. Expected End Product and Deliverables

- Easy to use interface.
- Templated diagrams to choose from.
- Free text edition of each part of the diagram.
- Securely store the user's data.
- Easy to install on every machine without too much pre-install requirements.
- Available on any platform (Linux, Windows, MacOS).

2. Specifications and Analysis

1. Proposed Design

So far, we have only gone over what technological, functional, and nonfunctional requirements that we want to deliver with this application. We plan on using Electron, which uses Chromium to make and render Node.JS, HTML, and CSS applications for users. We decided this would be good to use instead of a web application so that is easily usable, and we don t have to worry too much about security or user authentication when using the app. Electron also focuses on making well-crafted GUI applications using these web technologies. We have also gone over how we will use the Agile development practices to practices weekly and biweekly sprints throughout the semester.

2. Design Analysis

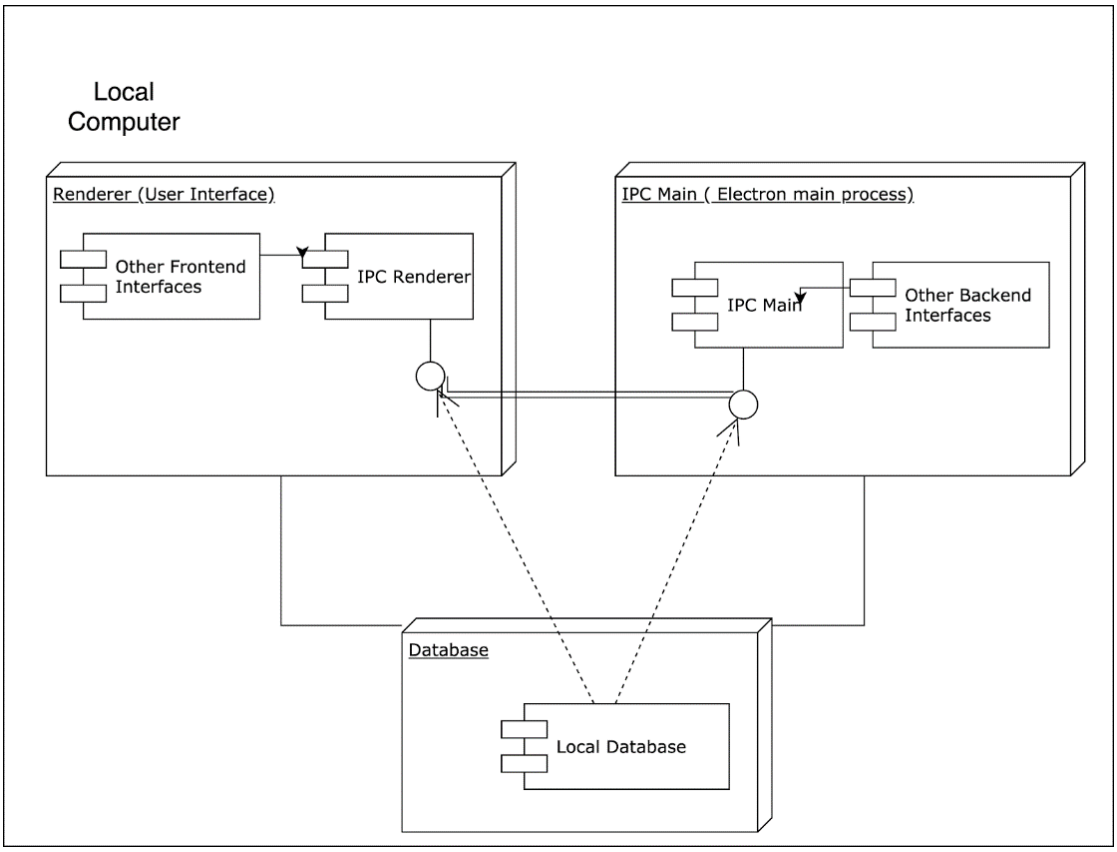
So far, no designs have been made other than high level diagrams and discussions regarding how we will be implementing our project.

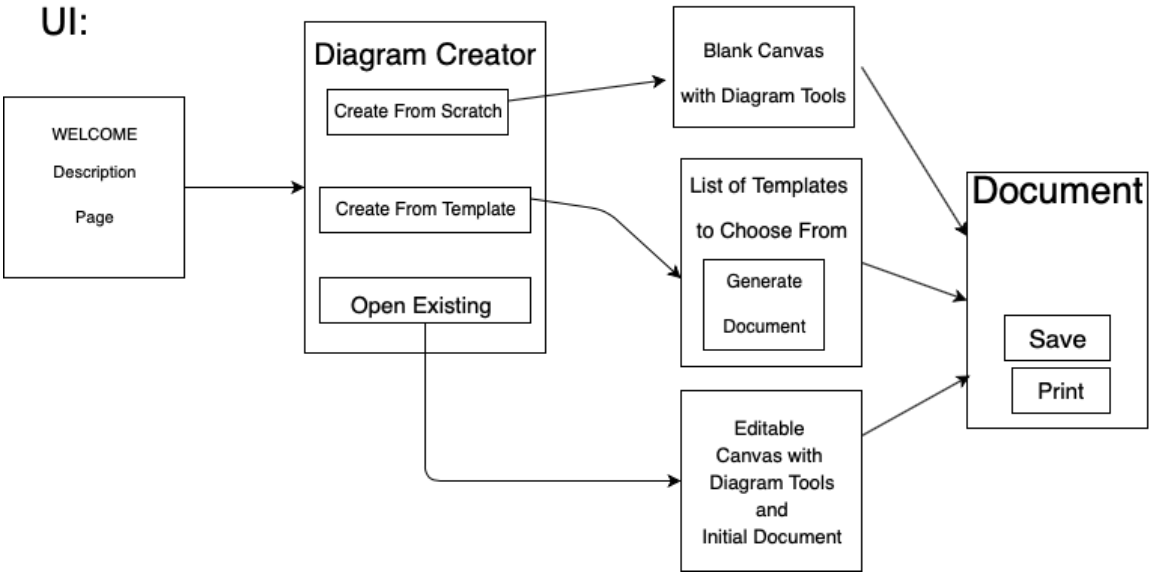
3. Development Process

We are using the Agile development process. We plan to have 2 major milestones in the semester where we will accomplish big feats in the project. We also plan on having weekly and biweekly sprints depending on how big they are, as well as using Trello to keep track of these sprints. We wanted to use Agile because of how efficient it is to use, and it was suggested by our faculty advisor Myra. Agile is very organized and allows you to run a test-driven development throughout our project time. Using sprints, you can easily lay out the status of your projects and allows you to only take work that you can handle at a give time, instead of having constant ongoing story cards going on at once.

4. Design Plan

Diagram:





3. Statement of Work

3.1 Previous Work And Literature

We will be modeling the look of our charts based on some existing software that is available, however we have not yet started this portion of the project so much is still to be discussed regarding this. We have been provided literature by our client/advisor regarding the relevant background knowledge needed for understanding the basics of Assurance Recipes. We will be utilizing past iGEM projects as reference, due to the availability of all previous projects.b

3.2 Technology Considerations

We decided early on that we wished to develop a NodeJS based application. After some discussion, we decided that developing an application that runs on the Electron framework would be more advantageous when compared to a standard web app. Using Electron, we would still be able to develop using NodeJS, but we would also not have to worry about a database and the required security for user credentials and logins. This decision was made after considering the necessary security measures that must be in place to comply with the standards set by the European Union.

3.3 Task Decomposition

In order to solve the problem at hand, it helps to decompose it into multiple tasks and to understand interdependence among tasks.

3.4 Possible Risks And Risk Management

Include any concerns or details that may slow or hinder your plan as it is now. These may include anything to do with costs, materials, equipment, knowledge of area, accuracy issues, etc.

3.5 Project Proposed Milestones and Evaluation Criteria

What are some key milestones in your proposed project? Consider developing task-wise milestones. What tests will your group perform to confirm it works?

3.6 Project Tracking Procedures

We plan to utilize Trello and GitLab to track progress and utilize Slack for group communication.

3.7 Expected Results and Validation

With our project, we plan to have an application that is widely available and easy to use for people working in the field of synthetic biology. Our project will be utilized by people who work with our advisor/client initially and then be utilized by more in the field of synthetic biology. Our advisor works in the field of synthetic biology and will be able to provide validation of progress over the course of the project.b

4. Project Timeline, Estimated Resources, and Challenges

4.1 Project Timeline

We are in the early stages of discussing the schedule for our project. The basics of our timeline at this point are:

- Weekly meetings with our client/advisor every Monday.
- We plan to have two sprint periods between now and the end of the semester, roughly around mid November (week of the 18th possibly) and early/mid December.b
    - The first sprint will most likely be devoted to developing the rigid structure of the application.
    - The second sprint will most likely involve implementing a template for an assurance case.
- We started development the week of October 28th.
- We will have the basic skeleton of the project developed for our meeting on Nov. 4th.

- We plan to meet as a group to work as we see necessary.

4.2 Feasibility Assessment

The project will be a desktop application that allows the user to generate and edit Assurance Recipe diagrams. The most difficult portion of the project that we are anticipating, is the development related to generating the charts and also have them be aesthetically pleasing.

4.3 Personnel Effort Requirements

We are just beginning the early stages of development for our project now, but we anticipate to spend 10-15 hours per week during the sprints of our project.

4.4 Other Resource Requirements

We do not have any known resource requirements for our project at this time.

4.5 Financial Requirements

We do not have any relevant financial requirements for our project.

5. Testing and Implementation

Testing is an **extremely** important component of most projects, whether it involves a circuit, a process, or a software library

Although the tooling is usually significantly different, the testing process is typically quite similar regardless of CprE, EE, or SE themed project:

1. Define the needed types of tests (unit testing for modules, integrity testing for interfaces, user-study for functional and non-functional requirements)
2. Define the individual items to be tested
3. Define, design, and develop the actual test cases
4. Determine the anticipated test results for each test case 5. Perform the actual tests
6. Evaluate the actual test results
7. Make the necessary changes to the product being tested 8. Perform any necessary retesting
9. Document the entire testing process and its results

Include Functional and Non-Functional Testing, Modeling and Simulations, challenges you ve determined.

1. Interface Specifications

– Discuss any hardware/software interfacing that you are working on for testing your project

2. Hardware and software

– Indicate any hardware and/or software used in the testing phase

– Provide brief, simple introductions for each to explain the usefulness of each

3. Functional Testing

Examples include unit, integration, system, acceptance testing

4. Non-Functional Testing

Testing for performance, security, usability, compatibility

5. Process

– Explain how each method indicated in Section 2 was tested

– Flow diagram of the process if applicable (should be for most projects)

6. Results

– List and explain any and all results obtained so far during the testing phase

- – Include failures and successes
- – Explain what you learned and how you are planning to change it as you progress with your project
- – If you are including figures, please include captions and cite it in the text

  • This part will likely need to be refined in your 492 semester where the majority of the implementation and testing work will take place

-**Modeling and Simulation**: This could be logic analyzation, waveform outputs, block testing. 3D model renders, modeling graphs.

-List the **implementation Issues and Challenges**.

6. Closing Material

6.1 Conclusion

Summarize the work you have done so far. Briefly re-iterate your goals. Then, re-iterate the best plan of action (or solution) to achieving your goals and indicate why this surpasses all other possible solutions tested.

6.2 References

This will likely be different than in project plan, since these will be technical references versus related work / market survey references. Do professional citation style(ex. IEEE).

6.3 Appendices

Any additional information that would be helpful to the evaluation of your design document.

If you have any large graphs, tables, or similar that does not directly pertain to the problem but helps support it, include that here. This would also be a good area to include hardware/software manuals used. May include CAD files, circuit schematics, layout etc. PCB testing issues etc. Software bugs etc.