# Open Source .NET Blog

DESIGN DOCUMENT

34
Cylosoft
Selim Mohamed
PJ DeBisschop, Cole Mulligan, Sebastian Kazun, Thien Nguyen
sdmay20-34@iastate.edu
sdmay20-34.sd.ece.iastate.edu

Revised: 11/3 v2

# Executive Summary

## Development Standards & Practices Used

Practices: Testing, Refactoring, shared codebase, simple design

## Summary of Requirements

- ASP.NET 4.8
- MSSQL
- Bootstrap 4.3
- CKEditor 4
- Microsoft Identity
- CRUD Blog Posts
- Comments (Approve / Deny)
- Categories (Filterable)
- Tags (Filterable)
- Site settings (Change theme)
- View most recent posts

## Applicable Courses from Iowa State University Curriculum

- COM S 309 Software Development Practices
- COM S 363 Intro to Database Management
- COM S 319 Construction of User Interfaces

## New Skills/Knowledge acquired that was not taught in courses

- .NET 4.8 Framework
- Microsoft SQL
- C#
- Bootstrap 4.3
- Microsoft Identity/Owin Authentication
- Model/View/Controller Architecture

# Table of Contents

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

# 1 Introduction

## 1.1 Acknowledgement

Thank you to Cylosoft for their technical and design support throughout the project.

## 1.2 Problem and Project Statement

General Problem Statement

- In the current state of the world there are a limited amount of open source .NET blogs for users to take advantage of. Those that exist are outdated and fall behind today's technology. Because of this Cylosoft would like their own blog platform which include additional features and allow them more control.

General Solution Approach

- A simple to use blog builder with an admin area to control content for readers.

## 1.3 Operational Environment

The ASP.NET server and database will be hosted through Microsoft's Azure cloud service.

## 1.4 Requirements

Functional Requirements

Normal User Area for viewing content

- A blog visitor shall be able to sort posts by most recent
- A blog visitor shall be able to filter posts by tags
- A blog visitor shall be able to filter posts by categories
- A blog user shall be able to view approved comments on posts
- A blog visitor shall be able to sign up/log in for an account
- A blog user shall be able to comment on posts
- A blog visitor shall be able to search for specific posts

Admin Area for controlling content

- An admin shall be able to Create, Read, Update, and Delete posts
- An admin shall be able to view new comments and approve them for others to see
- An admin shall be able to add posts to certain categories for users to filter by
- An admin shall be able to edit user roles
- Ad admin shall be able to change the theme

Non-Functional Requirements

- The software shall be accessible at any time
- The software shall not crash
- The software shall properly handle errors behind the scenes
- The software shall display content properly based on size and device
- The software shall utilize Microsoft Identity for security
- The software shall be designed in a way that allows easy changes to the codebase

UI Requirements

- The UI must be visually pleasing
- The UI shall be able to change theme

## 1.5 Intended Users and Uses

Our intention is to build an Open Source .NET blogging platform that allows Cylosoft to keep their customers informed about their activities. While also being flexible enough for other individuals or businesses to use as their own blog.

## 1.6 Assumptions and Limitations

Assumptions

- The end product will be open source.
- The end product will be able to support multiple users at a time.
- The end product will be viewable on various media devices

Limitations

- The length of time to produce the end product will not surpass May 2020.
- Enough database space to hold thousands of blog posts
- The cost of the product cannot exceed $0

## 1.7 Expected End Product and Deliverables

- Open Source .NET Blog Platform with source code
- Documentation
    - project plan, design document, database design, architecture design

Delivery date: May 1, 2020

# 2. Specifications and Analysis

## 2.1 Proposed Design

Our project will be built using .NET, so we have been experimenting with .NET web application development. Our design will use a properly managed and optimized database.

## 2.2 Design Analysis

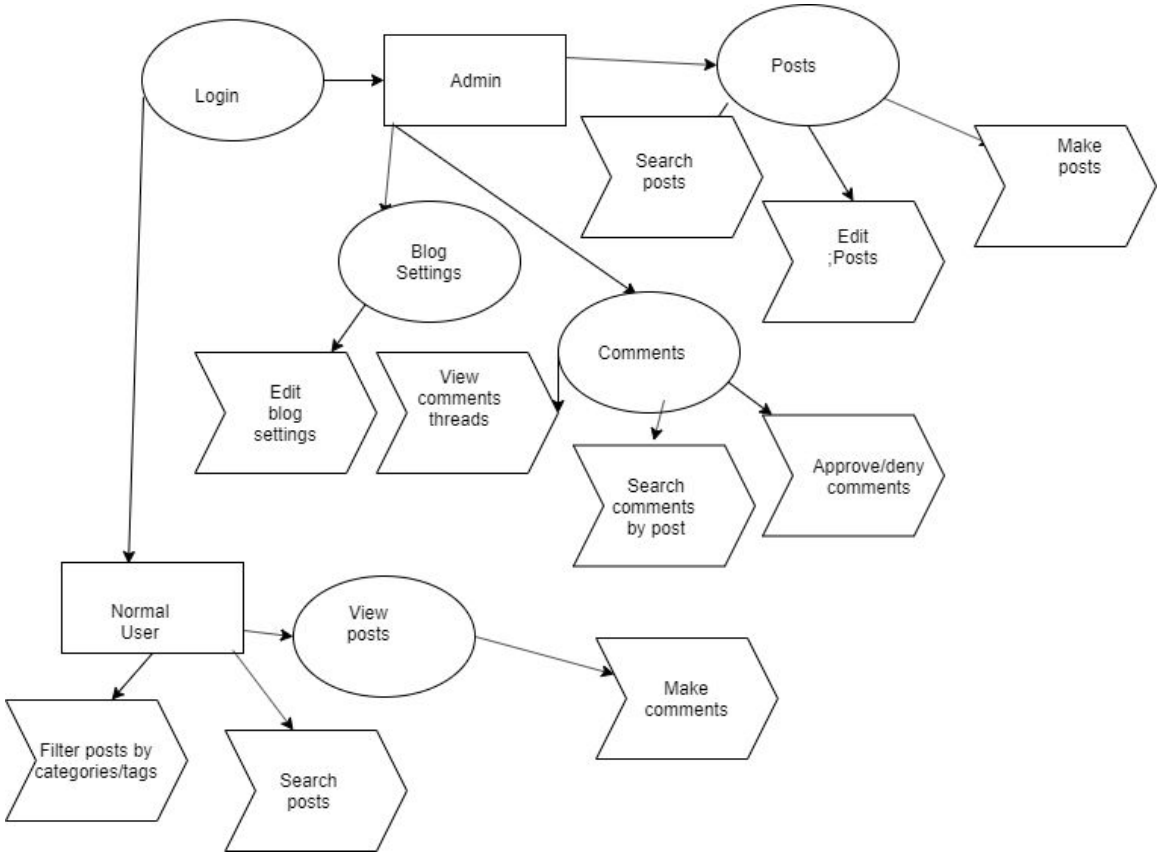


**Figure 2**

So far, we have began mocking up our user processes and researching .NET development. We plan to begin work with the User Flow Diagram in Figure 2 in mind. These are most of the requirements for our users to be able to operate their blogs and where in the website they can do each of these functions.

So far our team has spent time on learning the technologies needed to undertake our project. The technologies and languages we are using for our project can be found within our Executive Summary section and section 4.4.

Strengths

Weaknesses

## 2.3 Development Process

Our team will be following the Agile Development Process. We believe that adhering to the 12 principles will help immensely, allowing us to work flexibly and keep us on track. We will be using Trello as our agile board to help us assign tasks to the team. This will allow us to have a visual indicator on our progress for our current sprint and the effort of each teammate. Separating tasks into sprints allows us to more easily plan our work and make it manageable. We plan to release our first prototype by the end of the semester in December. From there we will work incrementally to further develop features and refine our project. We will end with a final product in early May.

## 2.4 Design Plan

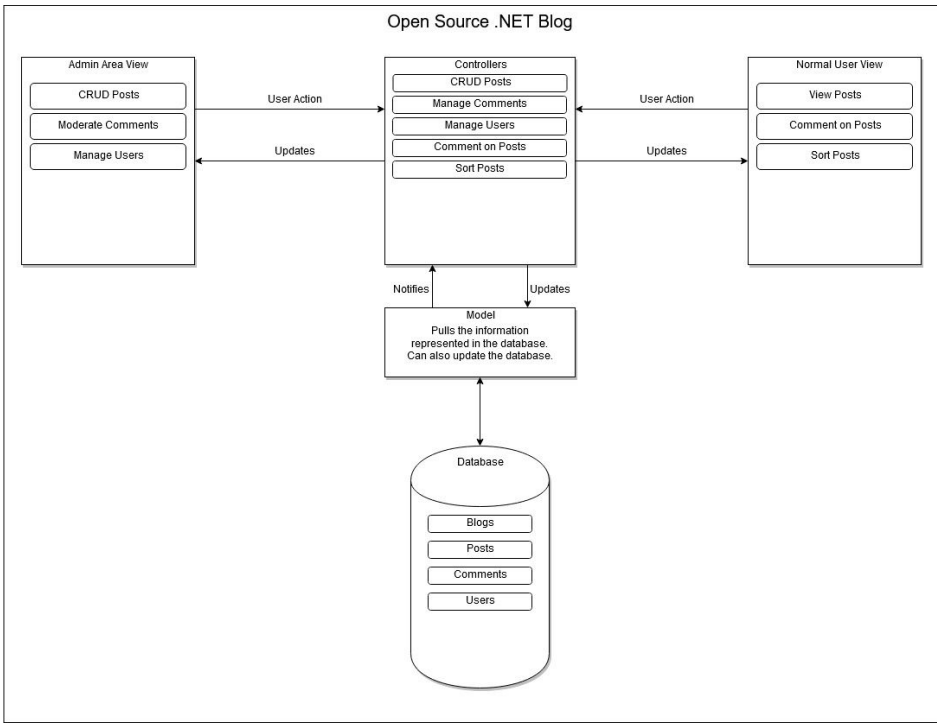


Figure 1

This is our first conceptual block diagram for how our Model View Controller system will function. You can start in either the Admin Area View on the left or Normal User View on the right, but both follow the same flow. Starting from the Admin Area View you will have options like CRUD Posts, a functionality allowing you to Create, Update, Read, or Delete any and all posts on the blog. After

deciding what action you would like to perform the VIEW sends a user action request to the controller module. Inside this main module are also others pertaining to different functionalities. Since we are doing CRUD Posts from the view we will also use the corresponding controller. This controller then updates the model, the model being our representation of the data contained inside the database. The model, aftering receiving the update, changing whatever needs to be changed will then notify the controller which will then update the corresponding view.

# 3. Statement of Work

## 3.1 Previous Work And Literature

BlogEngine.NET is the product that Cylosoft had us look at to get an idea of what we were building. This product is an open source blog builder with very similar features as to those required for our project. They offer a fairly easy setup and good design with clear methods for building the blog, with post management, comment management, user management, and a blog settings page. They also offer $10 theme packages for users to pick from.

Our project plans to offer a more attractive UI than similar products as well as incorporate up to date libraries. A point Cylosoft made was that current products aren't actively maintained so as technology advances, and current solutions just aren't up to date.

https://blogengine.io/

## 3.2 Technology Considerations

Our most prominent technology is developing with .NET, which is a popular option with good amounts of documentation making it easy for our team to pick it up quickly to begin work on the project. It also allows for simple server setup with many hosting services and integrates well with the other technologies in the project. C# will be used for handling a majority of our backend systems as it handles database transactions easily and has many examples performing similar features compared to our own solution.

Our database systems will run using MSSQL, another popular technology that stays in the Microsoft development ecosystem. Our account system is Microsoft Identity, which offers an easy to use way to register and login in users and user our MSSQL database to store basic account information securely.

The project will also utilize MVC for clear documentation and modular development purposes, as well as Bootstrap for responsive design.

A different design approach could be to not use MVC at all, in lieu of just using stand alone Razor pages. This is less clear code and simply wouldn't be as modular, and our projects benefits greatly for using modular pieces for things like posts between the admin system and the normal user views.

### 3.3 Task Decomposition

| |
|---|
| Database setup |
| ->Connect to database from application |
| ->setup.sql file containing table creation/initial values to execute from program |
| ->Add foreign keys |
| ->Look over "unused tables" for what we will use |
| ->Setup rights and designate them to their roles |
| Admin area (look, function, features) |
| Connect to Microsoft Identity |
| |
| Admin area finalizing design and begin implementation |
| Look, Features, Roles |
| ->Create posts |
| Implement Security Features/Database |
| ->Rights,Roles |
| ->Pull data from database into unit test project |
| ->Finish all models for database tables |
| ->Implement rest of the interface, repositories, & contexts |
| ->Create Units Tests |
| |
| Implement CKEditor 4 |
| ->Get CKEditor textarea working/visibile |
| ->Ability to POST to database |
| ->Ability to GET from database |
| |
| Login Page - Model, View, Controller |
| ->Implement Microsoft identity |
| ->Maybe upload some user info to our database for use with roles/rights? |
| |
| Database working (Able to add new user on login/change roles) |
| |
| Look into incorporating bootstrap |
| Look into Razor page themes |

### 3.4 Possible Risks And Risk Management

Our team is still new to many of the technologies and have not done complicated development in the system so there could still be knowledge gaps to overcome. However, active development is underway and our team is actively increasing getting more comfortable with development.

### 3.5 Project Proposed Milestones and Evaluation Criteria

1. Post creation
    a. Styled text
    b. Viewable to non-admins
    c. Include images
2. Changing blog theme
3. User management
    a. Login/Register
    b. Role assignment

We plan to implement unit testing for our backend system as well as several acceptance tests with Cylosoft.

### 3.6 Project Tracking Procedures

We are currently using an excel document to plan and keep track of our weekly progress. The Weekly Plan document outlines the week, the task to be done, who it is assigned to, completion percent, and shows which parts are dependent on each other. For next semester we plan to be using a Gantt chart to plan out our project and track its progress as we go, as it provides more visual information compared to our current setup.

### 3.7 Expected Results and Validation

Our desired outcome is to have a fully functional Open Source Blog up and running by the end of May. We hope to achieve all functionality that we have outlined throughout this document.

At a High Level we will confirm our solutions through acceptance testing: having our project contacts review and provide feedback on what needs to be adjusted. We will also revisit our requirements and make sure each one has been met.

# 4. Project Timeline, Estimated Resources, and Challenges

## 4.1 Project Timeline

| TASK NAME | START DATE | DAY OF MONTH* | END DATE | DURATION* (WORK DAYS) | DAYS COMPLETE* | DAYS REMAINING* | TEAM MEMBER | PERCENT COMPLETE |
|---|---|---|---|---|---|---|---|---|
| **First Sample Project** | | | | | | | | |
| Meet with Cylosoft and faculty | 9/10 | 10 | 9/13 | 4 | 4 | 0 | ALL | 100% |
| Gain ASP.NET knowledge | 9/16 | 16 | 10/25 | 40 | 40 | 0 | ALL | 100% |
| Map out database tables | 9/16 | 16 | 9/25 | 10 | 10 | 0 | PJ | 100% |
| Look into blog engine .NET | 9/16 | 16 | 10/25 | 40 | 40 | 0 | ALL | 100% |
| Microsoft Identity | 9/16 | 16 | 10/25 | 40 | 40 | 0 | ALL | 100% |
| **Second Sample Project** | | | | | | | | |
| Database setup | 10/20 | 20 | 11/1 | 13 | 12.61 | 0.39 | PJ | 97% |
| Initial MVC Project Setup | 10/22 | 22 | 10/30 | 9 | 9 | 0 | Sebastian, Cole, Thien | 100% |
| Connect Microsoft Identity | 10/22 | 22 | 11/15 | 25 | 18.75 | 6.25 | Sebastian | 75% |
| Database Integration | 10/30 | 30 | 11/15 | 17 | 3.4 | 13.6 | PJ | 20% |
| **Third Sample Project** | | | | | | | | |
| Login Page | 11/1 | 1 | 12/14 | 44 | 4.4 | 39.6 | Sebastian | 10% |
| Admin Page | 11/1 | 1 | 12/14 | 44 | 4.4 | 39.6 | Cole | 10% |
| Implement Security Features | 11/1 | 1 | 12/1 | 31 | 1.55 | 29.45 | PJ | 5% |
| Account Features | 10/29 | 29 | 12/14 | 47 | 0 | 47 | Thien | 0% |
| Implement CKEditor | 10/29 | 29 | 12/14 | 47 | 9.4 | 37.6 | PJ | 20% |


This project time goes until the end of this semester. This is because we will need to see where we are at at the end of the semester before developing a plan and timeline for then.

## 4.2 Feasibility Assessment

The end goal of our project is to have an open source blog application. It will be available for free on GitHub for anyone to download and use for themselves. Some challenges we expect to face our the implementation of all the desired features from Cylosoft and ensuring they work properly.

Another potential issue is making sure this application is truly "open source" and that other potential users are able to setup and use the application fairly easily.

### 4.3 Personnel Effort Requirements

| Task | Personnel Effort |
|---|---|
| Initial MVC Project Setup | Preliminary research into .NET MVC project structure and setup, as well as incorporating Razor pages and Microsoft Identity. Ensuring this is done properly, knowledge of the MVC architecture and how it is used will be essential to making the project. |
| Connection to Database | Look into the different ways of incorporating a database into a .NET application, such as the repository pattern. Perform an initial implementation and build out a testing project to verify it works as intended. |
| Design and Functionality of Admin Page | Build out a list of all the required functionality for the Admin page. Build a more detailed list of the step by step process each functionality will need in order to function properly. Create a mock up UI and begin implementation. |
| Testing | Test everything as it is being built. This essentially doubles everyone's work load. After implementation of a feature we will need to build out a test method to confirm our change works as intended. |

### 4.4 Other Resource Requirements

N/A

### 4.5 Financial Requirements

A hosting platform for the database and website is the only financial resource required.

## 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.

## 5.1 Interface Specifications

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

## 5.2 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

## 5.3 Functional Testing

Examples include unit, integration, system, acceptance testing

## 5.4 Non-Functional Testing

Testing for performance, security, usability, compatibility

## 5.5 Process

- Explain how each method indicated in Section 2 was tested
- Flow diagram of the process if applicable (should be for most projects)

## 5.6 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.